NPS52-89-036

NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

A GRAPHICS WORKSTATION FIELD ARTILLERY FORWARD OBSERVER SIMULATION TRAINER

by

William Thomas Drummond, Jr.

and

Joseph Paul Nizolak, Jr.

June 1989

Thesis Advisor: Michael J. Zyda

Approved for public release; distribution is unlimited.

Prepared for:

Director, United States Army
Combat Developments Experimentation Center
Fort Ord, CA. 93941

NAVAL POSTGRADUATE SCHOOL
Monterey, California

Rear Admiral R. C. Austin
Superintendent

Harrison Shull
Provost

This report was prepared in conjunction with research conducted for the United States Army TEC. The work was funded by the Naval Postgraduate School and the United States Army TEC.

This thesis is issued as a technical report with the concurrence of:

REPORT DOCUMENTATION PAGE

REPORT SECURITY CLASSIFICATION CLASSIFIED	1b RESTRICTIVE MARKINGS
SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution is unlimited
DECLASSIFICATION / DOWNGRADING SCHEDULE	
ERFORMING ORGANIZATION REPORT NUMBER(S) 352-89-036	5 MONITORING ORGANIZATION REPORT NUMBER(S)
NAME OF PERFORMING ORGANIZATION val Postgraduate School	6b OFFICE SYMBOL (If applicable) 52
7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School	
ADDRESS (City, State, and ZIP Code) terey, CA 93943-5000	7b ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000
NAME OF FUNDING / SPONSORING ORGANIZATION Naval Postgrad-e School and USATEC	8b OFFICE SYMBOL (If applicable)
9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER ATEC 44-87	
ADDRESS (City, State, and ZIP Code) NPS: Monterey, CA 93943 USATEC: Ft. Ord, CA 93941	10 SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO	PROJECT NO	TASK NO	WORK UNIT ACCESSION NO

TITLE (Include Security Classification)

RAPHICS WORKSTATION FIELD ARTILLERY FORWARD OBSERVER SIMULATION TRAINER

PERSONAL AUTHOR(S)

mmond, William T., Jr. and Nizolak, Joseph P., Jr.

TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
ter's Thesis	FROM _____ TO _____	June 1989	103

SUPPLEMENTARY NOTATION The views expressed in this thesis are those of the author do not reflect the official policy or position of the Department of ense or the U.S. Government

COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Graphics workstation, Field Artillery, forward observer, simulation, trainer, Digital Message Device

BSTRACT (Continue on reverse if necessary and identify by block number)

Today's forward observers need a low cost, realistic training system that fully prepares them for operations in any area of potential lict. We present a graphics workstation method of training Field Artillery forward observers to call for and adjust indirect fire. system uses the dynamics and flexibility of computer graphics to simulate mobile observers and targets operating in a three ensional environment. We produce three dimensional terrain from Defense Mapping Agency (DMA) Level 1 Digital Terrain ation Data (DTED). The program depicts a functionally accurate, on screen Digital Message Device (DMD), the same device ard observers use to input missions. To allow use in Field Artillery operations, we convert the DMA terrain files from graphic coordinates to the Military Grid Reference System (MGRS). We describe our simulator, the Forward Observer Simulation ner (FOST), listing its capabilities and features.

CPT Nizolak concentrated his efforts in the areas of 3D terrain and vehicle drawing algorithms and adapted the program to display A Level 1 DTED files in the MGRS. CPT Drummond's primary focus was on the operational aspects of the DMD. He provided a stic simulation of the DMD's capabilities that allow user input, change and transmission of fire mission data. Both CPT Nizolak CPT Drummond designed the 3D observation post and projectile effects icons.

DISTRIBUTION / AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION Unclassified
NAME OF RESPONSIBLE INDIVIDUAL hael J. Zyda	22b TELEPHONE (Include Area Code) (408)646-2305
	22c OFFICE SYMBOL 52Zk

ORM 1473, 84 MAR 83 APR edition may be used until exhausted
All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

Approved for public release; distribution is unlimited.

A GRAPHICS WORKSTATION FIELD ARTILLERY FORWARD OBSERVER SIMULATION TRAINER

by
William Thomas Drummond, Jr.
Captain, United States Army
B.S., United States Military Academy, 1979

and

Joseph Paul Nizolak, Jr.
Captain, United States Army
B.S., United States Military Academy, 1979

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
June 1989

ABSTRACT

Today's forward observers need a low cost, realistic training system that fully prepares them for operations in any area of potential conflict. We present a graphics workstation method of training Field Artillery forward observers to call for and adjust indirect fire. Our system uses the dynamics and flexibility of computer graphics to simulate mobile observers and targets operating in a three dimensional environment. We produce three dimensional terrain from Defense Mapping Agency (DMA) Level 1 Digital Terrain Elevation Data (DTED). The program depicts a functionally accurate, on screen Digital Message Device (DMD), the same device forward observers use to input missions. To allow use in Field Artillery operations, we convert the DMA terrain files from geographic coordinates to the Military Grid Reference System (MGRS). We describe our simulator, the Forward Observer Simulation Trainer (FOST), listing its capabilities and features.

CPT Nizolak concentrated his efforts in the areas of 3D terrain and vehicle drawing algorithms and adapted the program to display DMA Level 1 DTED files in the MGRS. CPT Drummond's primary focus was on the operational aspects of the DMD. He provided a realistic simulation of the DMD's capabilities that allow user input, change and transmission of fire mission data. Both CPT Nizolak and CPT Drummond designed the 3D observation post and projectile effects icons.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

We thank Dr. Michael J. Zyda for all his support and encouragement throughout the development of FOST. He gave us guidance when we needed it and respected our ability to set our own goals. He is a great teacher and advisor.

We greatly appreciate Mr. Michael Tedeschi and The United States Army Test and Experimentation Command for their support and funding of our research efforts.

Most importantly we thank our lovely wives, Cheryl and Mary, and our sons, Michael and Billy, for their patience and support during the past twenty-one months at the Naval Postgraduate School. Without them our achievements would not be possible.

I. THE NEED FOR A BETTER FORWARD OBSERVER TRAINER

A. CHALLENGES OF FUTURE BATTLEFIELDS

The forward observer (FO) on the AirLand battlefield faces challenges like no forward observer before. Fluid battle lines, rapidly changing situations, a highly mobile, numerically superior enemy, all dictate a need for the FO to plan quickly. These young soldiers must execute missions rapidly, on target, and while on the move. Instead of the classic infantry platoon in the open, their targets are mechanized forces swiftly moving over the battleground. The FO's ability to accurately bring suppressive fires to bear on this enemy is critical to the success of the maneuver element he supports.

B. CURRENT FORWARD OBSERVER TRAINING METHODS

Throughout the US Army there are two standard forums for training our FOs for the next battle. One is actual live fire and the other is live fire simulation, principally on the Training Set Fire Observation (TSFO). Let's examine how each of these methods are meeting the training requirements for the AirLand Battle.

1. Live Fire

There has probably not ever been a field artilleryman in the world who did not love the thrill of calling for indirect fire from the hill. The challenge of accurately bringing fire to bear on a simulated enemy and seeing the flashes of "steel on target" amounts to a key test of whether or not we can fulfill our mission to support the

ground gaining arms. There is no argument that a live fire exercise is beneficial training for the next battle. During a live fire exercise, we get an opportunity to train the entire fire support system: guns, fire direction centers and FOs. The FOs get a hands-on experience of putting steel on the targets that their Fire Support Team (FIST) NCO or Chief designates.

Unfortunately, and especially in these days of budgetary constraints, we seldom get to live fire enough. Live fire is expensive in all classes of supply, as well as in time and training area requirements. There are few, if any, FOs who can consider themselves ready for combat based on their live fire training. Live fire has other training shortfalls that we must consider when we look at alternatives to prepare for the AirLand Battle. As we said earlier, our potential enemies are highly mobile. We must train our FOs to plan and dynamically adjust fires on moving targets. This skill is a repetitive weakness during National Training Center (NTC) exercises and no wonder, the car bodies and dumpsters on the impact areas only move when they receive direct hits! Safety constraints on our impact areas also cause training constraints by reducing the types of munitions the FO can request. Those same safety constraints, in most circumstances, eliminate his ability to fire danger close missions. Calling fire missions on the move and danger close, as he will in the next conflict, is rarely, if ever, practiced. Finally, and most obviously, the local impact area is not our future battlefield. Using live fire training, our FOs are not becoming familiar with the terrain on which they will actually fight.

2. The TSFO.

The TSFO is a computer synchronized array of slide projectors that gives forward observers a two dimensional view of terrain. By providing the ability to call for

and adjust indirect fires on a screen, the TSFO picks up where live fire leaves off. Because of relatively low resource requirements, the TSFO allows us to train our FOs virtually everyday. This training makes actual live fire exercises more cost effective by drilling the FOs on basics before a round is fired. The TSFO offers a controlled training environment for the FIST NCO. He can concentrate on his soldiers and on their training weaknesses. FIST NCOs can work on any problems FOs are having in target location or call for fire procedures, without having to worry about how accurate the battery is shooting that day. The FO determines a target location and the rounds appear on the screen exactly at that location. The TSFO also enhances training by allowing FOs to train on simulations of several projected battlegrounds, something live fire just cannot do.

The TSFO, however, is not without some serious deficiencies that we must consider. Regardless of efforts to make the 35mm slide show seem real, the bottom line is the TSFO is two dimensional. The greatest challenge to the FO is adjustment for range and he just cannot train well on the TSFO due to its inability to allow the FO to use his depth perception. The TSFO prohibits simulation of shooting on the move and only displays an approximate 6x6 km training area. Even a cursory glance of AirLand Battle doctrine tells us that we do not stay in any 6x6 area very long and shooting on the move is how we fight. Development of a 6x6 km area for TSFO use is expensive in terms of money, time and manpower. The TSFO site manager obtains the terrain elevations for his slides, by the inherently inaccurate method of map spotting. Very few terrain depictions are available and none are available for hostile areas such as Warsaw Pact nations or the Middle East. Compounding the deficiency of the small area of operations, is the limitation of moving targets to a total of eight vehicles. Trainers must plan these vehicles a day prior to scheduled training and cannot

change the pre-programmed routes during training. A final problem is that a TACFIRE interface is not part of the fielded TSFOs, though many enterprising units have come up with their own modifications that allow the TSFO to communicate with TACFIRE. We are an automated artillery and our training systems should reflect that fact.

There are modifications to the TSFO that will enhance its training value. The project management office for the TSFO is monitoring the fielding of a Ground/Vehicular Laser Locator Designator (G/VLLD) enhancement that replicates a G/VLLD in appearance and function. Another modification the project management office is considering is the ability to interface with multiple Digital Message Devices so that FOs can conduct simultaneous fire missions.

These modifications to the TSFO are good and do enhance its training value. However, because of its inherent two dimensional effect, stationary observers and limited availability of battlefield terrain depictions, the TSFO will continue to fall short in training our FOs for the AirLand Battle. Preparing for the AirLand Battle requires training techniques which safety and cost prohibit during live fire and for which simulation systems must provide a forum. We must close the current gap between what the TSFO provides and what we in the Field Artillery need.

3. A Better Way

Advancements in three dimensional visual simulation software can provide a better system for training FOs for the next battle. We use the Naval Postgraduate School's Moving Platform Simulator as our base model in order to create the Forward Observer Simulation Trainer (FOST) [Ref. 1]. Its basic requirements are a low cost, off the shelf, graphics workstation and the program code. We integrate the capability to display any Defense Mapping Agency (DMA) Digital Terrain Elevation Data

(DTED) Level I files with realistic forward observer missions. Our goal is a cost effective system that capitalizes on these software advancements and available data to satisfy the need left by current training methods. We believe a system like FOST holds many answers to the question of how we can best train our soldiers to be ready when our country calls.

II. HISTORICAL DEVELOPMENT

A. INTRODUCTION

Recent developments in high performance graphics workstations and digital terrain elevation databases have allowed students at the United States Naval Postgraduate School (NPS) to produce some very good and relatively inexpensive moving platform simulators [Ref. 1]. There is, however, a trade-off between realism and performance; essentially, the more realistic and detailed the simulation, the slower the simulator. In order to apply these developments to a training forum, we must achieve a satisfactory level of realism while attaining acceptable performance.

We use NPS's Moving Platform Simulator (MPS) [Ref. 2] as our base model to create a prototype forward observer trainer. By improving on the current simulator's realism and creating an accurate training interface, we move the simulators from basic research into the training applications arena. In order to better understand our base model, the Moving Platform Simulator, we will first review the system and its evolution.

B. FIBER-OPTICALLY GUIDED MISSILE (FOGM) SIMULATOR

Graphics students at NPS developed the FOGM simulator [Ref. 3] in June 1987 on a Silicon Graphics, Inc. IRIS 3120 graphics workstation. The FOGM simulator was the first in a series of simulators ultimately leading to the design of MPS. This simulator presented the user with a moving three-dimensional view of terrain from the perspective of a missile. The creators used a 10 kilometer x 10 kilometer area of Fort Hunter-Liggett, California for their terrain database. In addition to the terrain, the simulator was also capable of displaying moving vehicles to which the user assigned initial headings and speed. Since the IRIS 3120 does not have the hardware to support real-time, double-buffered, hidden surface elimination, the creators used a

scanline Painter's algorithm for all drawing. This algorithm sorts polygons from farthest away to closest to the viewer's position and then draws them in that order. This method ensures that distant objects do not obscure objects closer to the viewer and that the program draws vehicles after displaying the terrain. [Ref. 3]

C. VEHICLE SIMULATOR (VEH)

Research students completed work on VEH in December 1987 [Ref. 3]. VEH also ran on the Silicon Graphics, Inc. IRIS 3120 graphics workstation. It used the same terrain and vehicle drawing algorithms as FOGM and allowed for real-time selection and control of ground vehicles [Ref. 3]. In order to improve performance, VEH used the scanline Painter's algorithm to only draw terrain that fell within the field-of-view .

D. FOGM/VEH NETWORKING SIMULATOR (FOGM/VEH NET)

FOGM/VEH NET was a networking simulator that linked FOGM and VEH [Ref. 4]. An Ethernet local area network linked the graphics workstations and permited simultaneous vehicle position updating between them. For example, a user operating a vehicle on a workstation running VEH could see the actions of a user operating a different workstation running FOGM.

E. VEH II VEHICLE SIMULATOR

The VEH II simulator, completed in June 1988, was the result of not only software enhancements to the VEH but also porting the VEH from the IRIS 3120 to an IRIS 4D/70G and an IRIS 4D/70GT. VEH II retained all the capabilities of the VEH simulator in addition to the modifications that allowed the simulator to run on the newer hardware and under the MEX and 4Sight [Refs. 5, 6, 7] window management systems. Enhancements to VEH II included popup menus for the user to select options, the ability to add vehicles to the simulator at any time, and an option to save

initial vehicle setups (i.e. relative positions of vehicles, speeds and headings) to a file for future simulations. [Ref. 1]

F. MOVING PLATFORM SIMULATOR (MPS)

Graduate students at NPS completed development of MPS from the FOGM and VEH II simulators, in December of 1988. By designing MPS on an IRIS 4D/70GT graphics workstation these students were able to take advantage of many hardware features in order to improve realism and performance. MPS allows the user to choose a 10 kilometer x 10 kilometer operational area from a 35 kilometer x 35 kilometer database. The terrain color scheme is variable and an efficient terrain drawing algorithm displays more terrain than earlier models by including distance attenuation. The program employs Z-buffering for hidden surface elimination. MPS also includes a lighting model which allows the user to adjust the month and hour of the day. By making these adjustments, the user sets the parameters for realistically lighted vehicles and terrain. The system enhances the FOGM missile with the ability to track, target, and destroy vehicles. MPS's collision detection scheme destroys colliding vehicles and missiles, rendering them inoperative. Broadcast networking in MPS permits multiple simulations to run on different IRIS 4D/70GT graphics workstations. [Ref. 1]

G. TERRAIN DATABASE

The United States Army Test and Experimentation Command at Fort Ord, California provided the terrain database that all the simulators use. The database is a special Defense Mapping Agency (DMA) database that consists of elevation and vegetation data in 12.5 meter increments in a 36 kilometer x 35 kilometer area encompassing Fort Hunter-Liggett, California. Each data point contains 16 bits. The three most significant bits are a vegetation code, which the simulators ignore, and the remaining 13 bits represent the elevation of the point measured in feet. The Moving Platform Simulator uses a 35 kilometer x 35 kilometer area with a resolution of 100

meters. At the time of this writing, an updated version of MPS is able to display the 12.5 meter resolution data. [Ref. 8]

III. CURRENT FOST TRAINING CAPABILITIES

To fight and win, FOs must accurately locate and fire on moving targets while they themselves are on the move. Our soldiers must be familiar with projected battlegrounds around the world. Requesting appropriate munitions to attack a target must be second nature. FOST can train FOs on the skills they need. Because of its menu and mouse driven environment, FOST is simple to operate. An on-screen users' manual is available at the start of the training session that reviews mouse and dial operation. To illustrate how FOST operates, let's look in on a typical Fire Support Team lead by a fictional Staff Sergeant (SSG) Smith. He and his FIST team are about to begin a training session using FOST.

A. INITIAL SETUP

SSG Smith creates a training scenario to fit his team's needs. He begins by selecting the terrain database, which could be any standard DMA Level 1 Digital Terrain Elevation Data (DTED) file, for the area of operations.* This is a very simple operation consisting of popup menu selections using a standard computer mouse. SSG Smith's selection causes loading and processing of the terrain data file. FOST now displays the database SSG Smith selected in a very familiar format, a two dimensional map with numbered Military Grid Reference System (MGRS) grid lines.

---

*A standard DMA Level 1 DTED file is one degree in latitude by one degree in longitude. It contains elevation data taken at 100 meter intervals. An area this size covers approximately 3600 square miles.

We use a standard map color scheme and key to distinguish elevations. A red box outlines the center 10x10 km area and another popup menu automatically appears (see Figure 3.1).

SSG Smith again uses a mouse click to select an option of using either the center area as his initial area of operations or to move the highlighted box to any other 10x10 km area on the map. All FOST menus also offer options to exit the program or to return to previous selection levels. After selection of the initial area of operations, FOST loads in the elevation data points for the 3D terrain and then displays this area as a full screen MGRS map. This display format allows easy placement of vehicles and observation posts (OP) (Figure 3.2). SSG Smith is now ready to input the tactical scenario.

Because FOST puts the trainer in charge, the initial tactical situation is SSG Smith's decision. A series of popup menus guide SSG Smith through placing friendly and enemy vehicles, stationary OPs, FISTVs and OH-58s, setting their initial locations, directions and speeds (see Figure 3.3). SSG Smith can choose to load situations he previously created or create new situations and have FOST save them to a file for later use. Storing tactical situations allows repetitive training to build strengths or correct any shortcomings of a previous session. The focus of the training is the trainer's call.

Supporting maneuvering forces while they close with and destroy the enemy is arguably the most important fire support mission. SSG Smith knows this and FOST allows him to realistically train his soldiers on fluid situations. Because he places friendly vehicles as well as enemy, his team can practice adjusting on the enemy

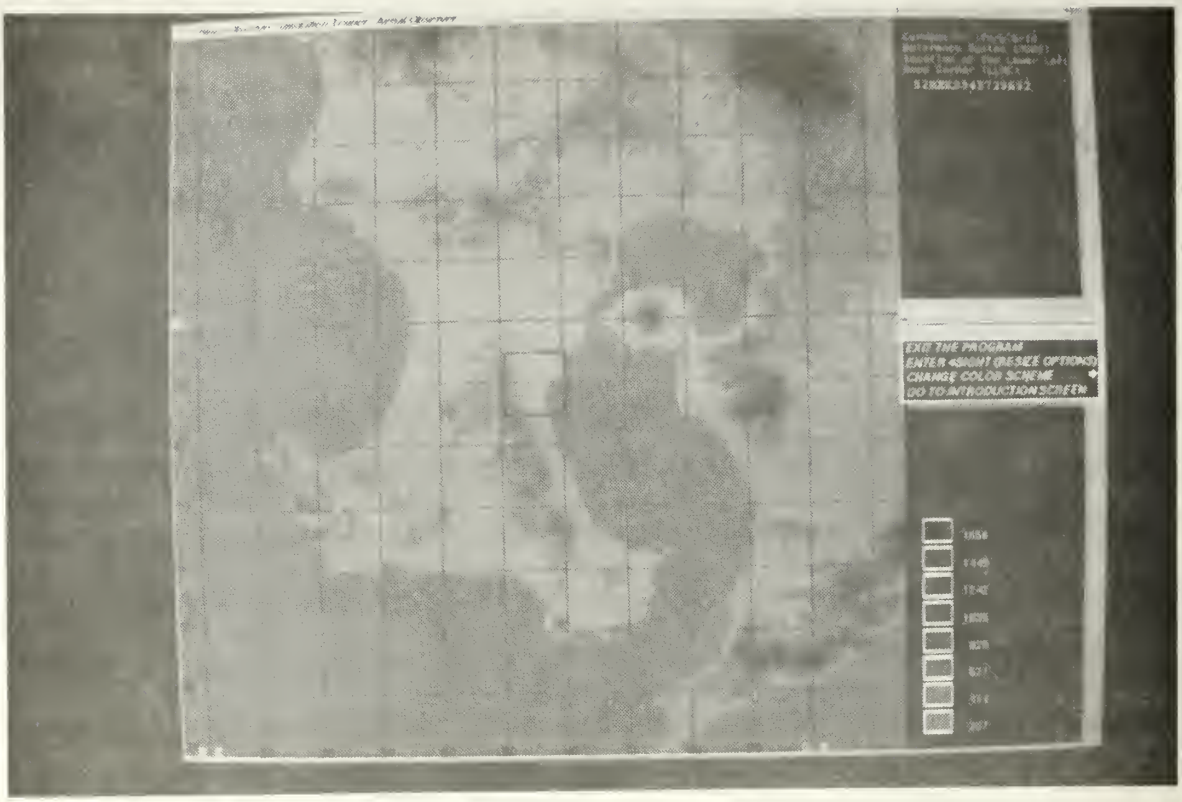


Figure 3.1 Full Database Map

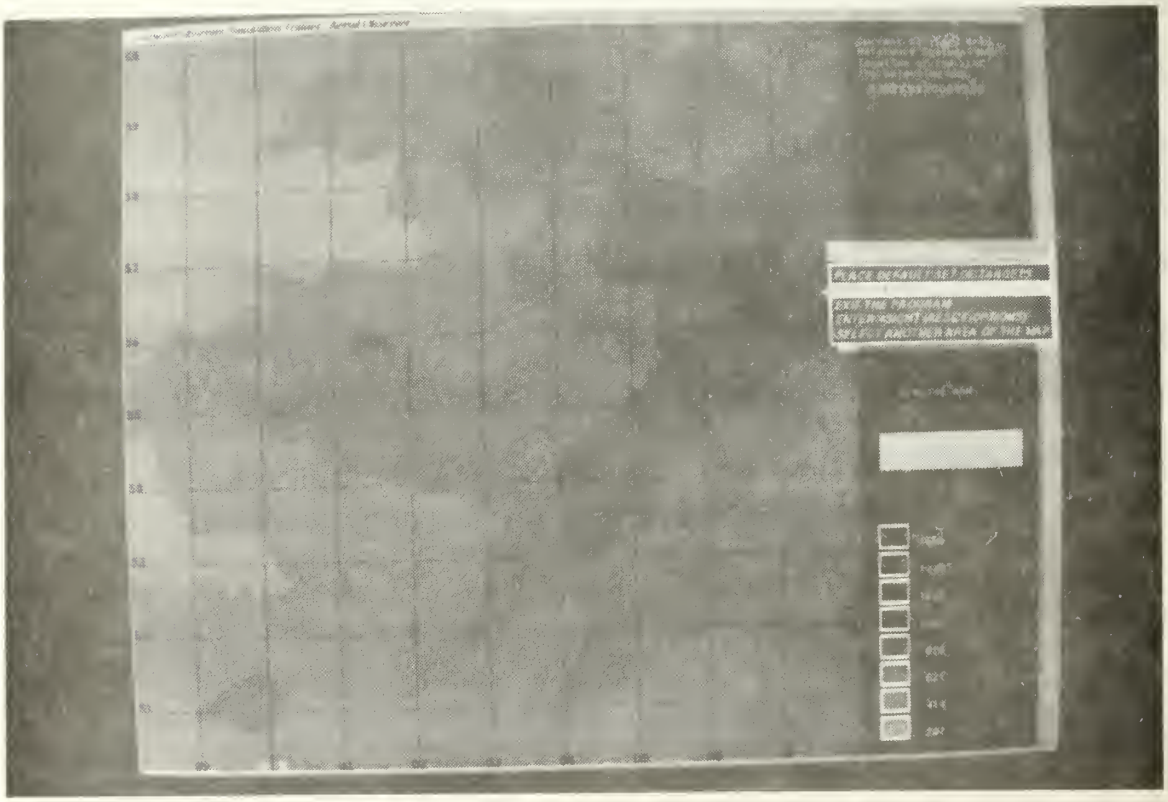


Figure 3.2 Initial Area Of Operations Map

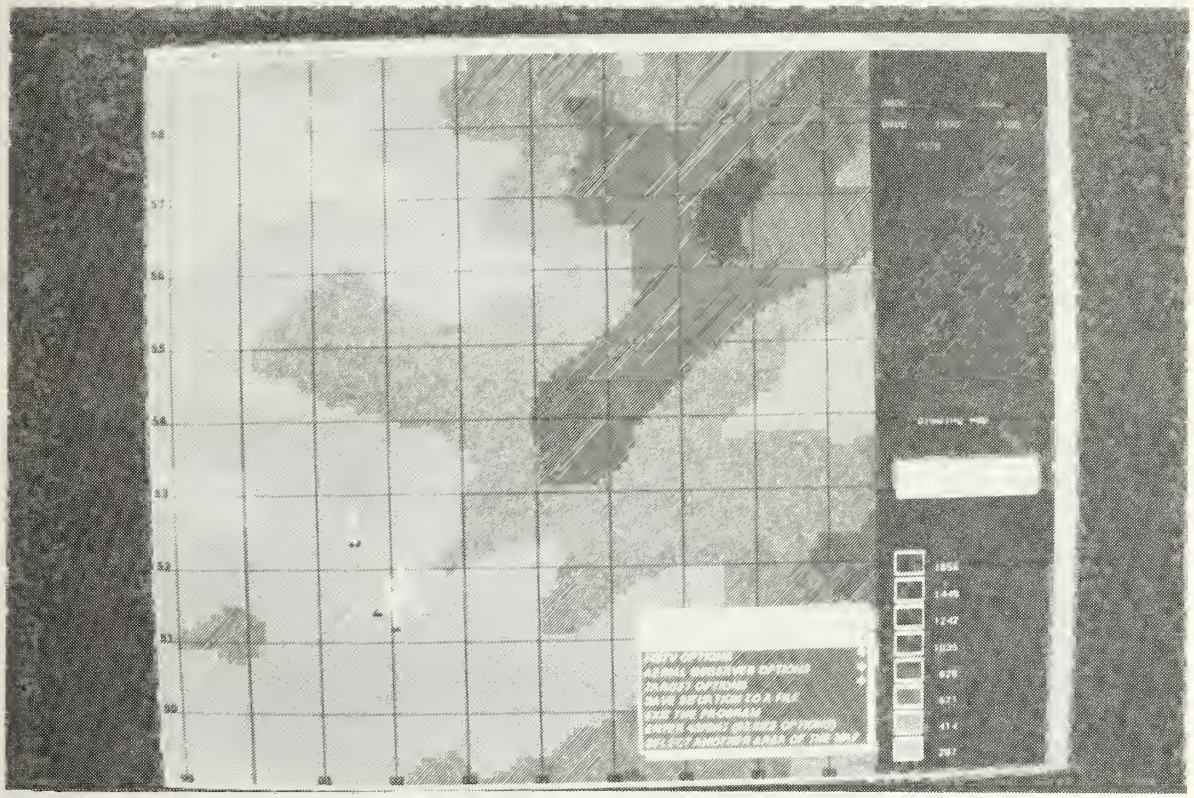


Figure 3.3 Tactical Situation

while they advance with their supported unit. The FO can see the friendly forces rather than imagine them on the impact area or on the TSFO.

B. CHALLENGE THE INDIVIDUAL

SSG Smith is an outstanding trainer and FOST supports his technique of challenging each team member based on their individual skill level. He has his new soldiers simulate occupying stationary OPs and directs them to engage the stationary targets on the battleground (see Figure 3.4). SSG Smith's more seasoned team members simulate occupying FISTVs, while the advanced FOs get to adjust from a scenario that simulates an aerial observer in an OH-58 (see Figures 3.5 and 3.6). The moving OPs are easier to control with two team members: one to drive/fly using a simple set of dials to regulate speed and direction, the other to operate the DMD (see Figure 3.7). SSG Smith challenges the FOs in moving OPs (FISTV and OH-58) with moving targets. He knows that with this type of individual training his team can get ready for Lieutenant Jones' FIST exercise, which will use FOST's networking capabilities.

C. TRAINING REALISM

"Live fire" training begins with occupation of the simulated observation post. SSG Smith supervises and assesses all the actions of his FIST team, while they fight the battle. The main window (see Figure 3.8) of the screen display shows a three dimensional depiction of the terrain that SSG Smith selected. Because FOST produces this terrain from DMA data, SSG Smith's FOs use their standard military maps to orient themselves and locate targets. Although FOs never get lost, FOST still provides reference information such as compass heading and current grid location in the right mar-

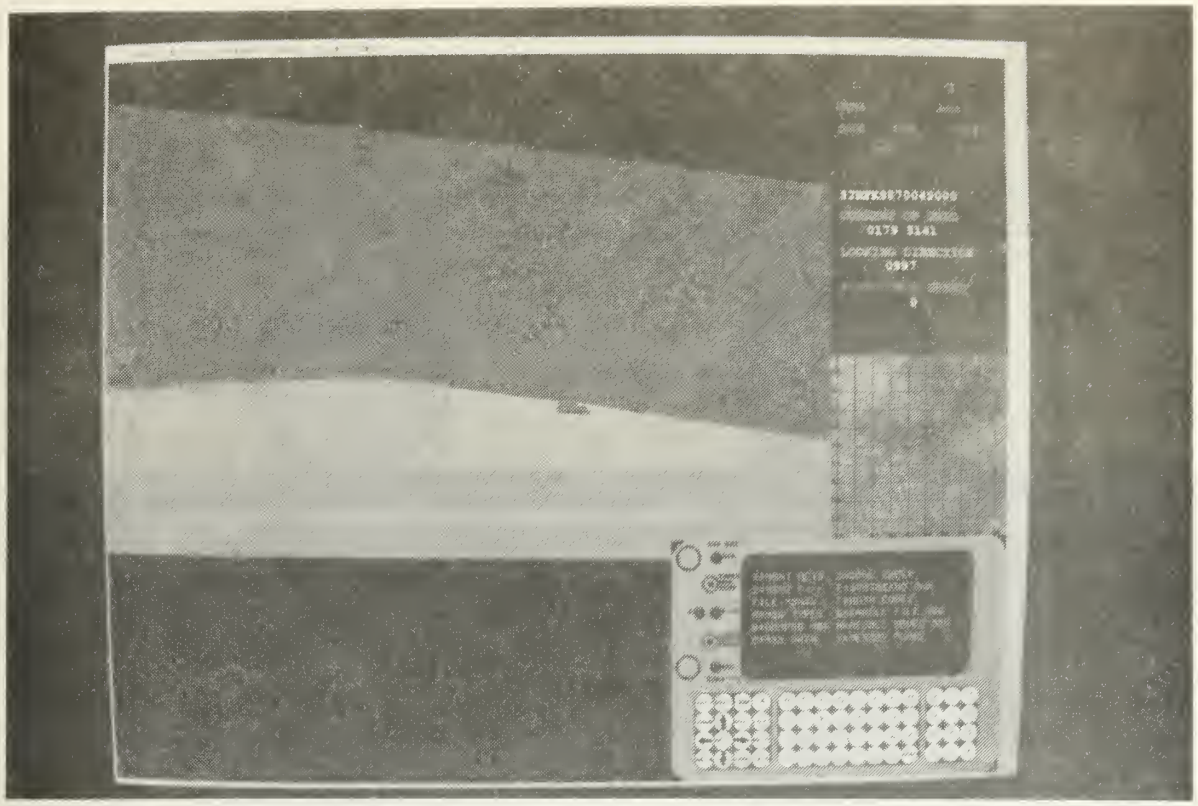


Figure 3.4 View From A Stationary OP

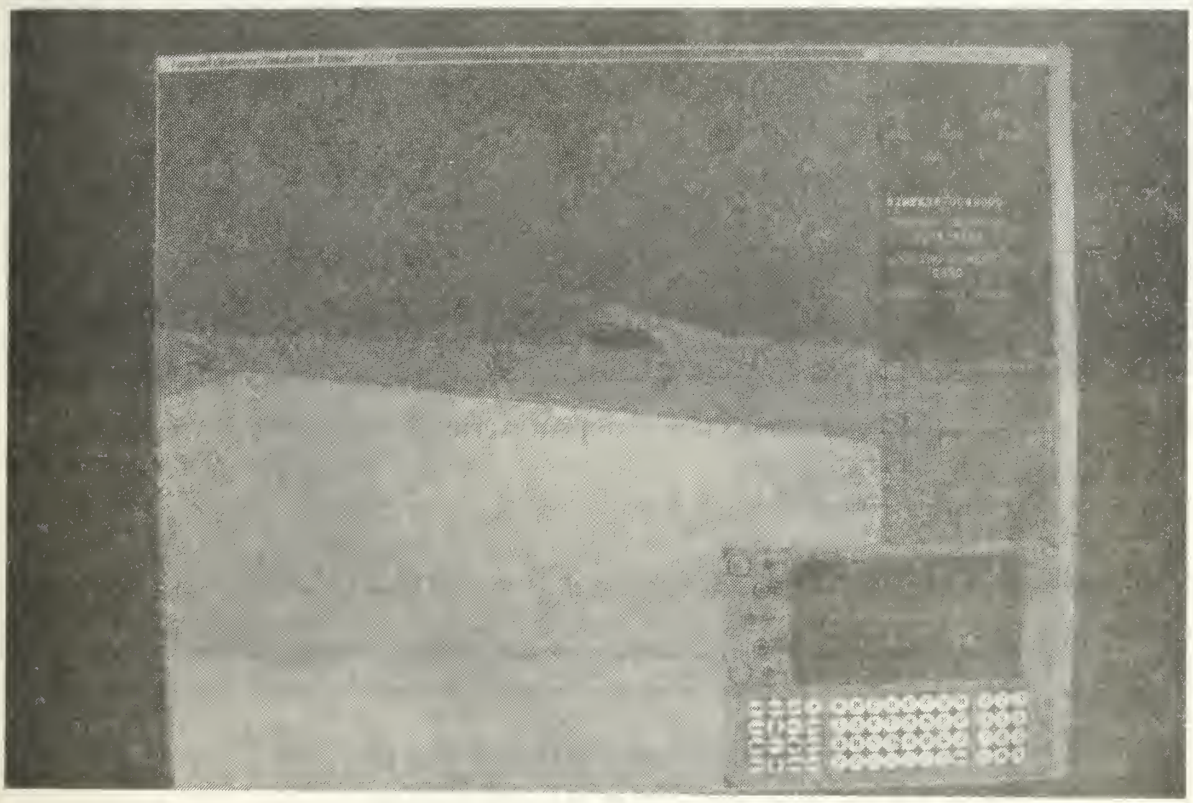


Figure 3.5 View From A FISTV

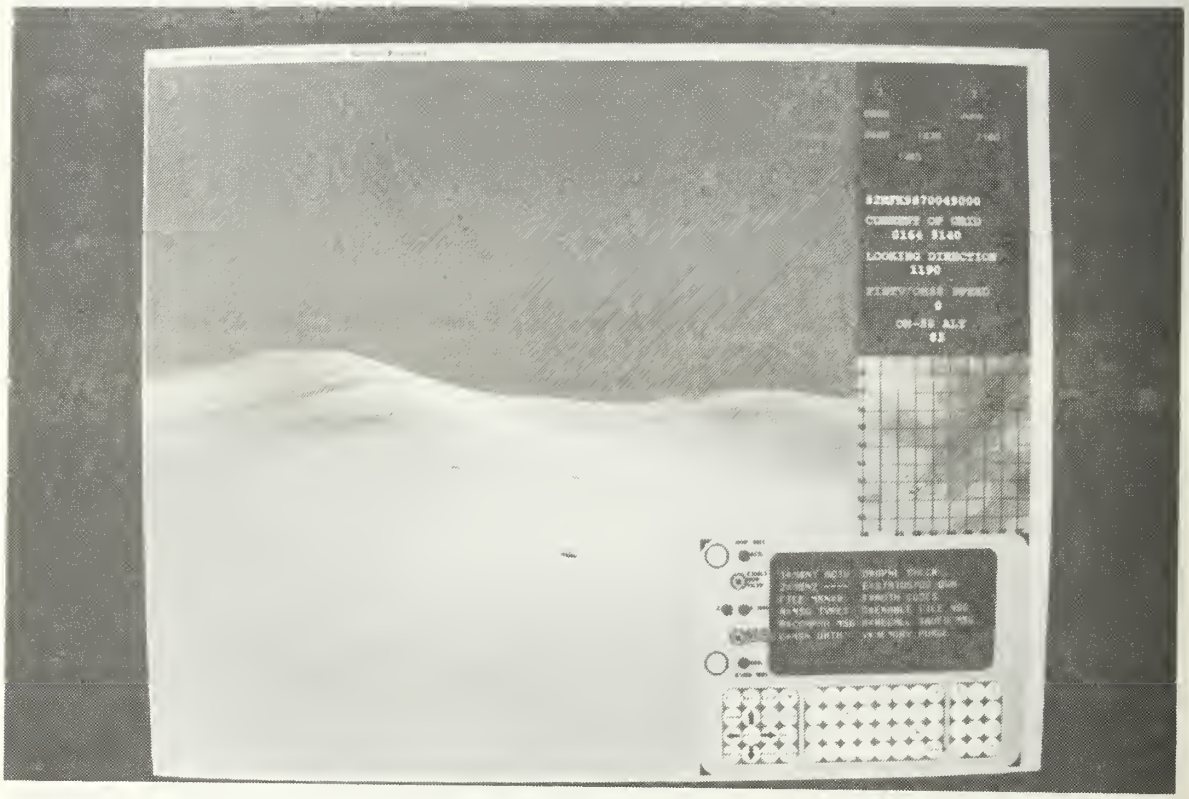


Figure 3.6 View From An Aerial Observer

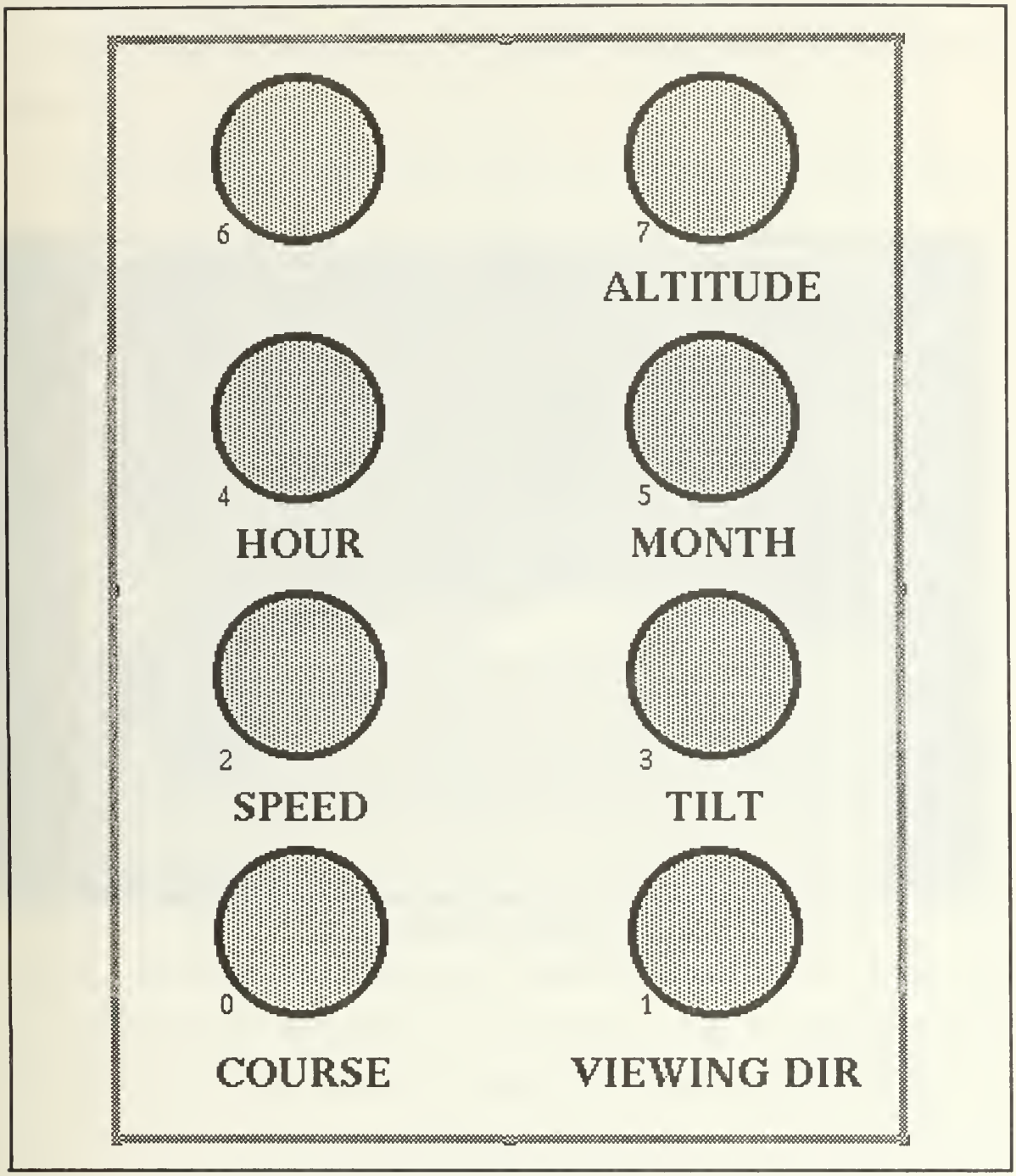


Figure 3.7 FOST Dial Controls

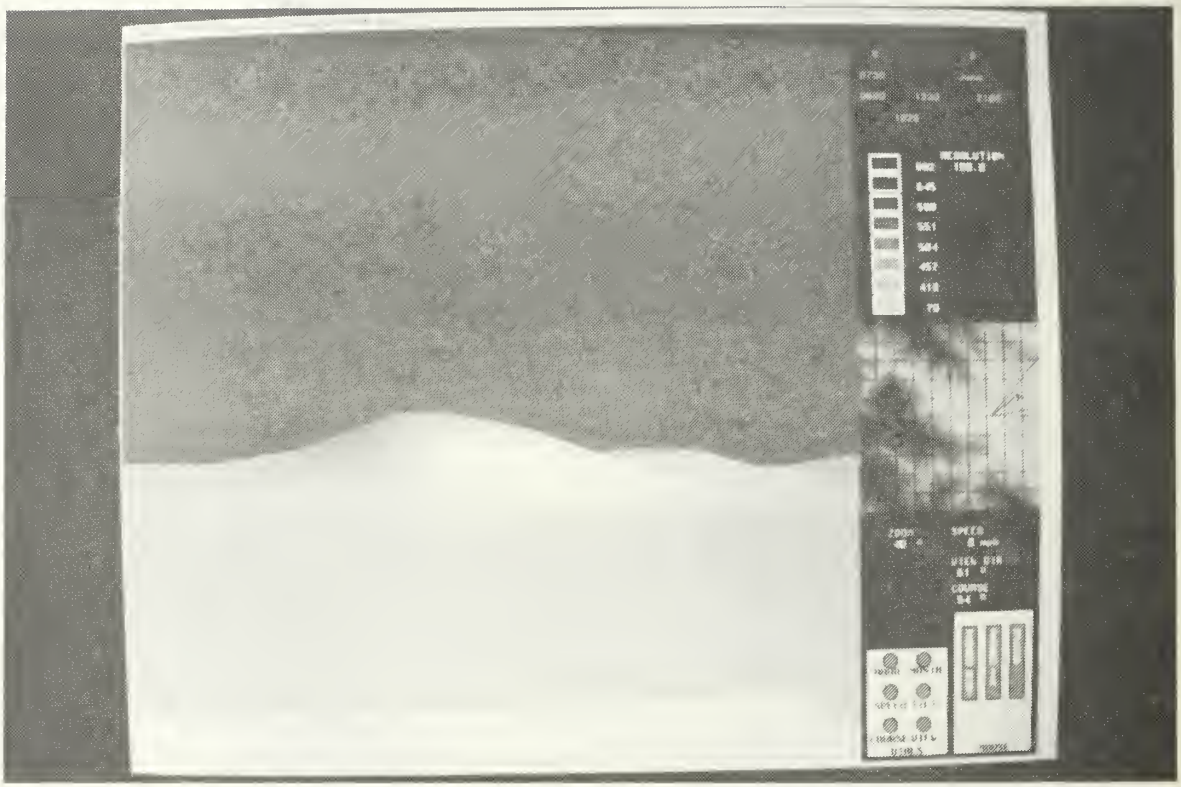


Figure 3.8 FOST Full Screen View

gin of the screen. The functional, on-screen DMD appears in the lower right corner (see Figure 3.9). FOs conduct fire missions using the DMD by "pressing" the on-screen keys using the mouse cursor. SSG Smith's team is now ready to engage targets.

FOST allows SSG Smith to train his team with a dynamic and realistic situation. The FOs input fire missions via the on-screen DMD to adjust fire or fire for effect. The team attacks targets using appropriate high explosive ammunition and receives clear feedback of a target destroyed when the adjustment is within 50 meters.* Knowing that an FO wouldn't be a real FO without his binoculars, FOST provides a magnified view, complete with reticle pattern, at the click of a mouse button (see Figure 3.10). SSG Smith can also change the area of operation, add more targets or switch assigned observation posts with popup menu selections.

D. FOST'S NETWORKING CAPABILITY

FIST is a team and should train as a team. FOST networking provides a team training capability. FOST uses a broadcast networking scheme where each program sends packets containing information about situation changes in its program as they occur. Other programs continuously check the network for packets from other workstations. Upon receiving a packet, each workstation updates its screen display with the information from other stations. The end result is that each workstation shows the sum of all the activity of each FOST program. Different users can see the same

---

*We currently have the following shell/fuze combinations: High Explosive/Point Detonating, High Explosive/Variable Time, High Explosive/Mechanical Time and Improved Conventional Munitions.

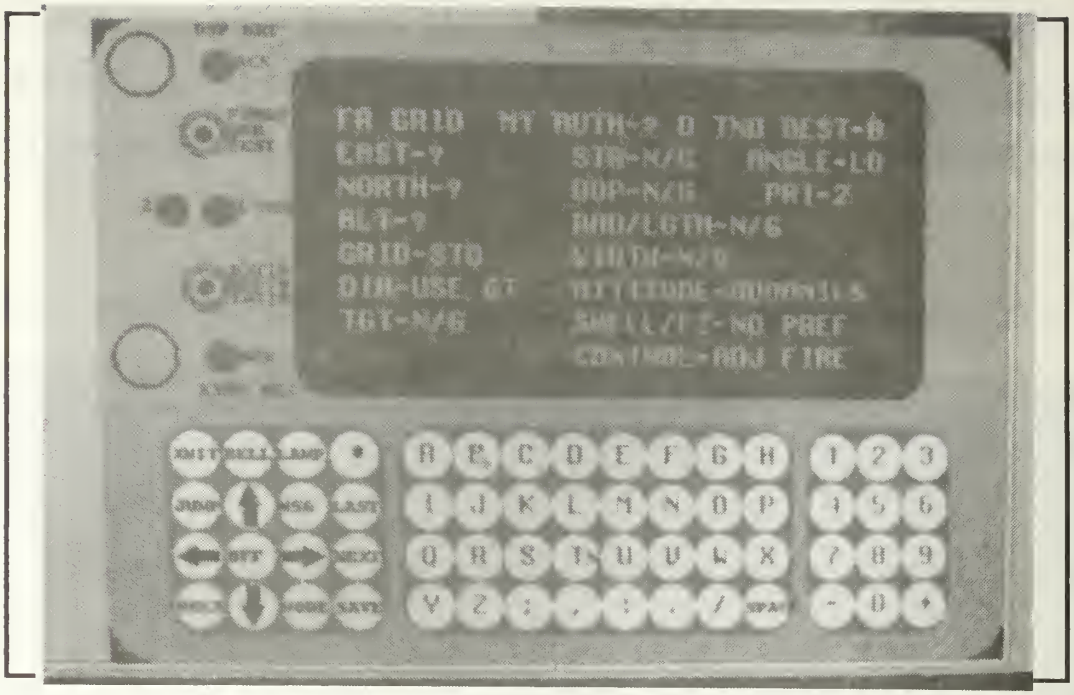


Figure 3.9 DMD Display

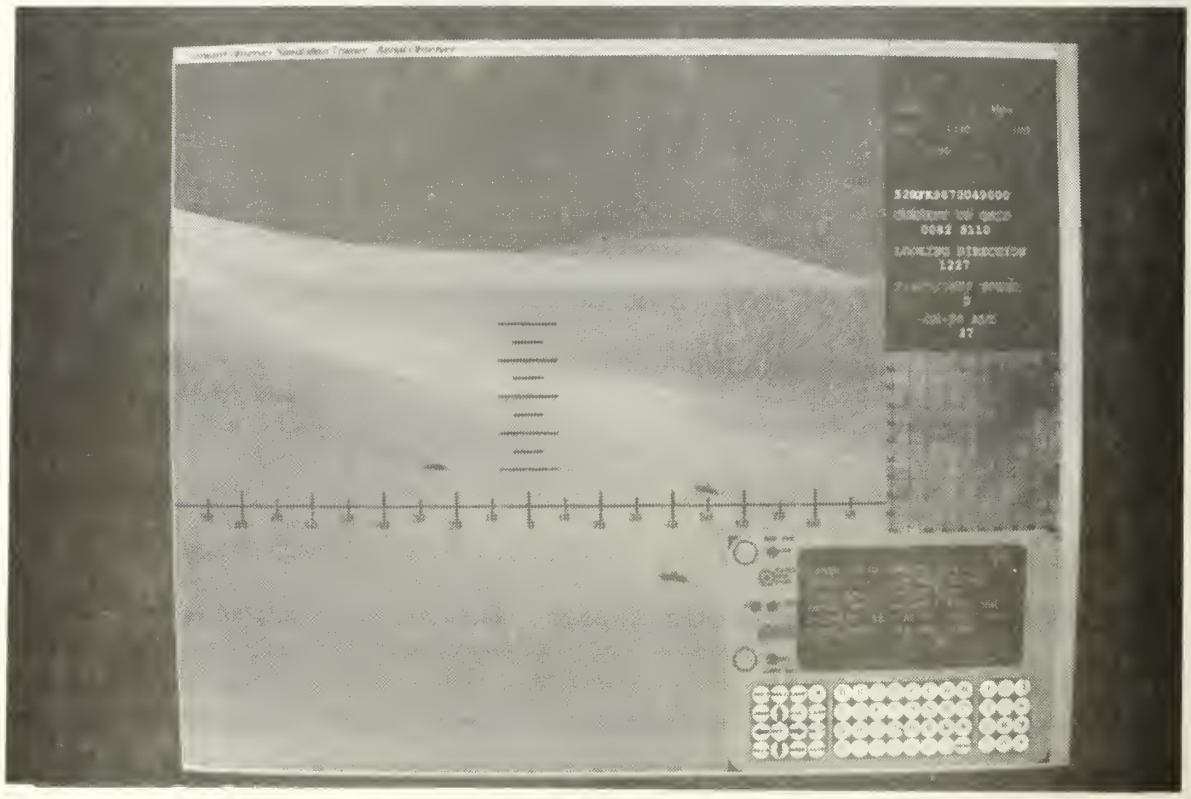


Figure 3.10 View With Binoculars

target from different angles and assist each other in target location and adjustment. An aerial observer (AO) can "pop up" from behind a hill, shoot a mission and receive effects on target information from a ground OP, while the AO returns to safety behind the hill. This networking capability allows FIST to train as a team and greatly enhances FOST's utility as a training asset.

E. CONCLUSION

Even in its current prototypical configuration, FOST offers innovative training in the hands of the unit level troop leaders. With FOST, trainers set the tempo for their individual soldiers and their units based upon those requirements that the trainers determine need the most attention. The features found in FOST are not duplicated in any other single Field Artillery training system in today's Army.

IV. FIELD ARTILLERY CONSIDERATIONS

A. THE DIGITAL MESSAGE DEVICE (DMD)

1. Description

A real DMD, officially known as AN/PSG-2A, is a small, lightweight, portable, two-way communications terminal. FOs use DMDs to transmit and receive digital messages between themselves and other fire support elements with digital devices via wire or standard radios (see Figure 4.1) [Ref. 9]. The messages an FO transmits and receives deal primarily with indirect fire support (e.g. fire missions and fire planning).

2. Accurate Design For Training Realism

Because a DMD is the FO's primary means of communication, we designed our on-screen DMD to resemble, as closely as possible, a real DMD (see Figure 4.2). FO's acquainted with DMDs find in FOST a device that is familiar in both appearance and function. Likewise, as FOs with little or no experience with digital devices learn how to operate the on-screen DMD, they are in fact learning the keying sequences and key positions of a real DMD.

To enter the data for a fire mission, the user places the mouse cursor over the appropriate DMD key and presses the middle mouse button. This action simulates the physical pressing of a real DMD's key. The middle mouse button is reserved for

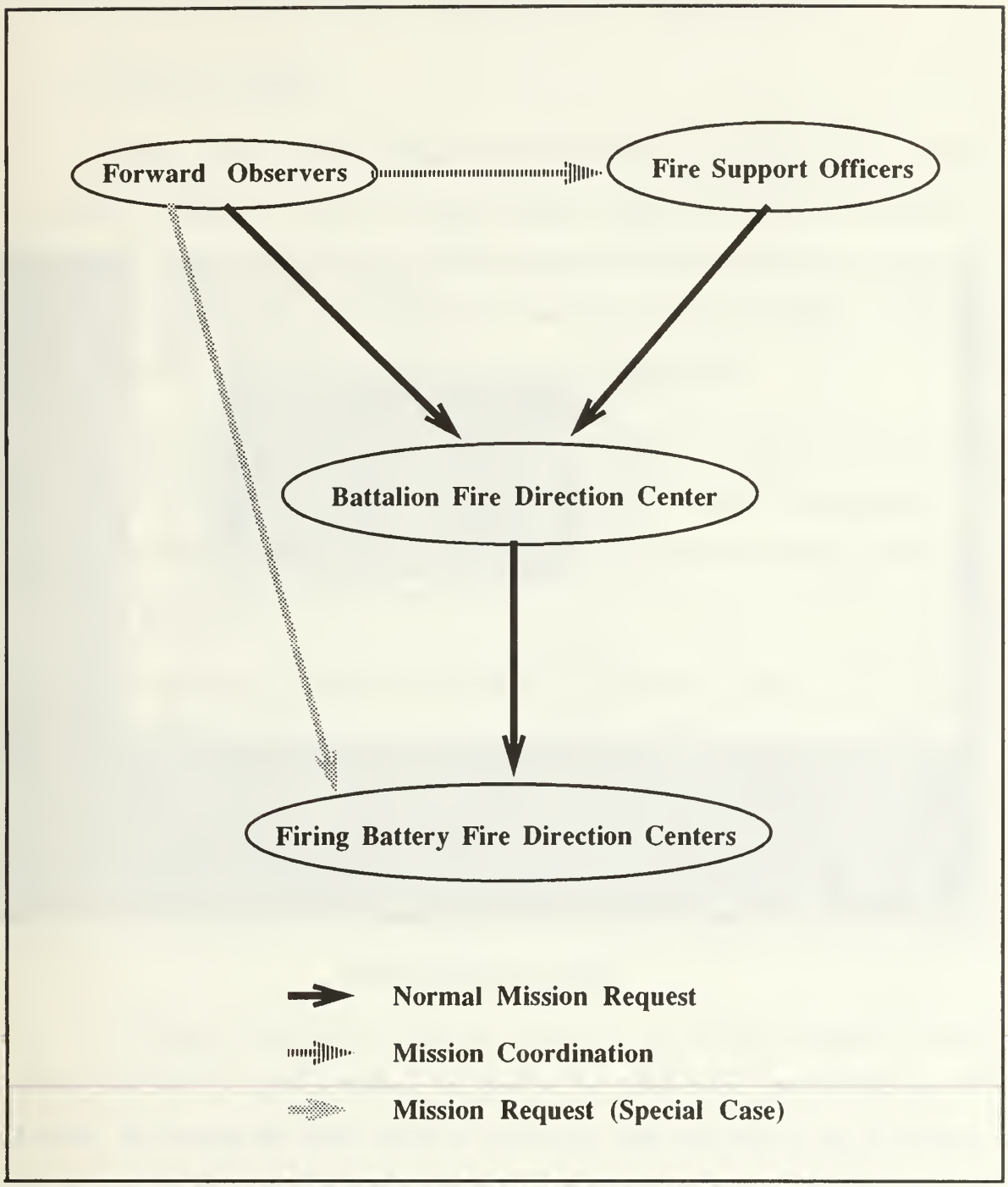


Figure 4.1 Fire Support Digital Communications Nets

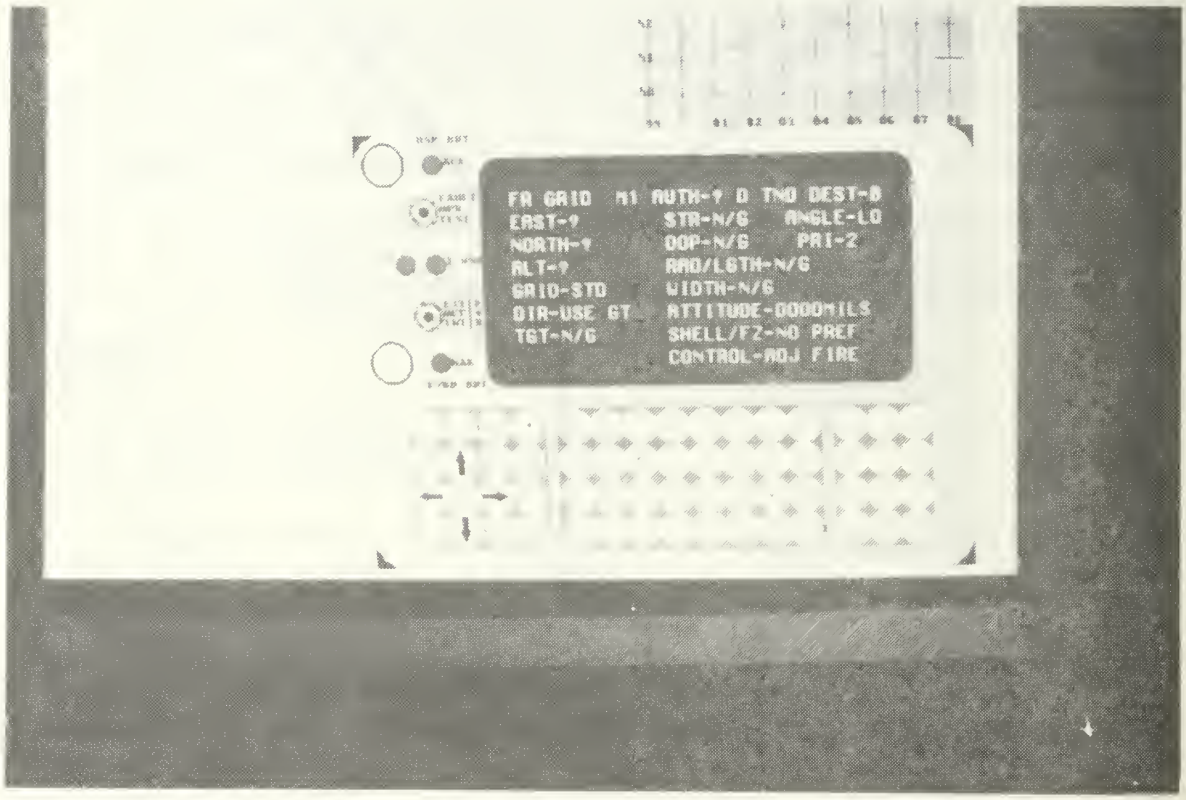


Figure 4.2 FOST DMD

and activates all the DMD operations (such as "pressing" the on-screen keys). All future references to "pressing" FOST's DMD keys refer to using the mouse.

3. FOST DMD Menus

There are four types of menus in the FOST DMD: opening, option, fill-the-blank and message summaries. Opening menus indicate to the FO the menu sequence that appears next and have question marks in fields which the FO will make entries during the sequence. There are two opening menus in our DMD: fire request grid mission (FR GRID) and subsequent adjustment (SUBQ ADJ).

Option menus present the FO with a list of selections. He can choose one by moving the DMD's cursor with the on-screen arrow keys to the desired option. He then presses the DMD's JUMP key to simultaneously make his selection and to proceed to the next menu.

Fill-the-blank menus allow the FO to enter mission dependent data such as target grid locations, directions and authentication values. Once the FO enters all the required information for a fill-the-blank menu, the DMD registers the data and automatically advances to the next menu. Before the FO enters last element in the data field, he can use the DMD's left arrow key to backspace and erase data he just entered.

Message summaries are the last menus in a particular sequence. These menus look like the opening menus, but have all the fields filled with the data the FO entered. By moving the DMD cursor to a particular field and pressing the JUMP key, the FO can return to the menu for that field and change the entry. For the cases of grid missions and subsequent adjustments, when the FO is satisfied with the data on

the summary, he presses the DMD's XMIT key to "transmit" the data. Once the FO transmits his data, FOST performs the necessary calculations and graphics routines to depict the artillery round and its terminal effects.

4. Using The DMD In FOST

In FOST, the only type of fire request currently available is the grid mission. Our DMD is tailored to support only that mission and the subsequent adjustments following the initial round. We built in only the minimum essential menus necessary to accomplish that support.

The first menu on the DMD's screen is the same as the one that appears on a real, initialized DMD (see Figure 4.3). This menu functions as a base menu and appears whenever the FO presses the DMD's MODE key. The only working option on this menu in FOST is "A=MSG TYPES". This option produces a menu that presents all the available message types in the DMD.

The Message Types menu shows 20 preformatted messages (see Figure 4.3). Each of these messages represent a means for the FO to enter data for various missions and operations. Since we support only the grid mission and subsequent adjustments, only two of the 20 message options are currently operational: "B=FR GRID" and "F=SUBQ ADJ". Both of these options generate a sequence of menus to prompt the FO for the necessary mission information.

There are several menus/messages common to both the FR GRID and SUBQ ADJ sequences: authentication (AUTH), direction (DIR), shell/fuze combination, fire control, and two other short messages that indicate a successful transmission. The authentication and direction menus are both fill-the-blank type menus (FOST accepts

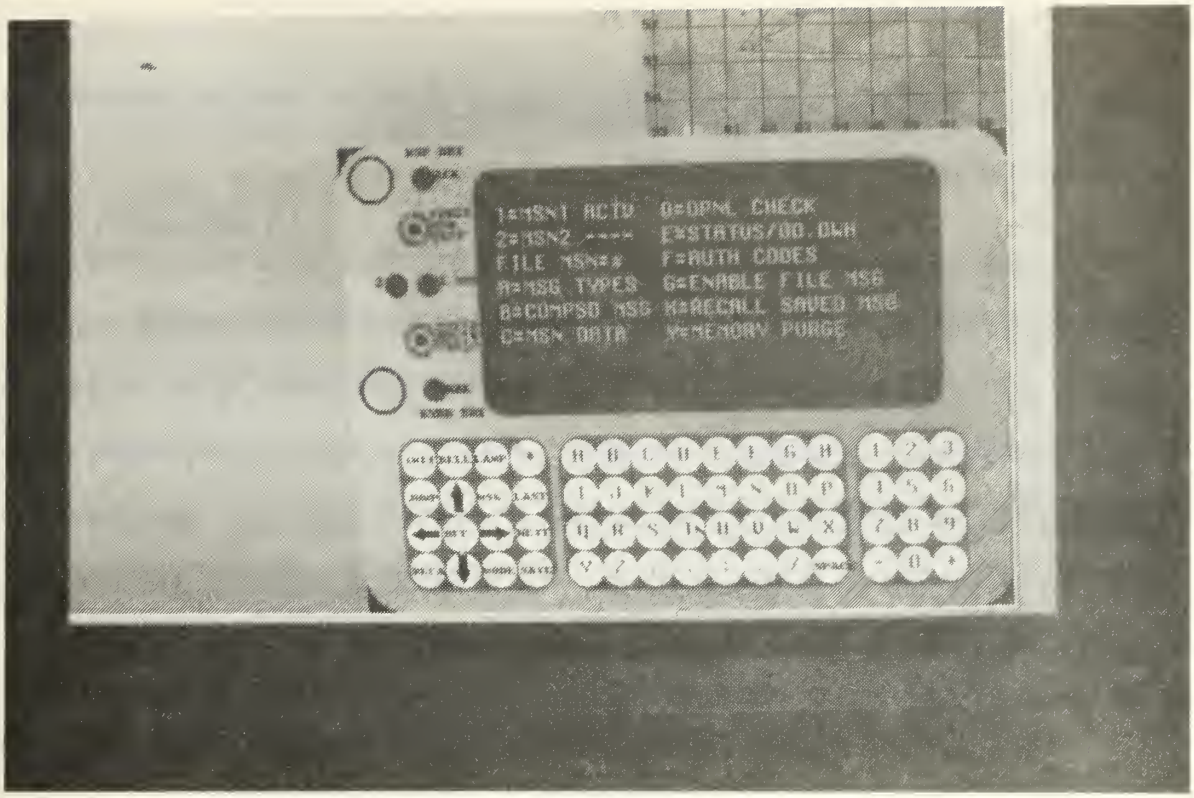


Figure 4.3 First DMD Menu

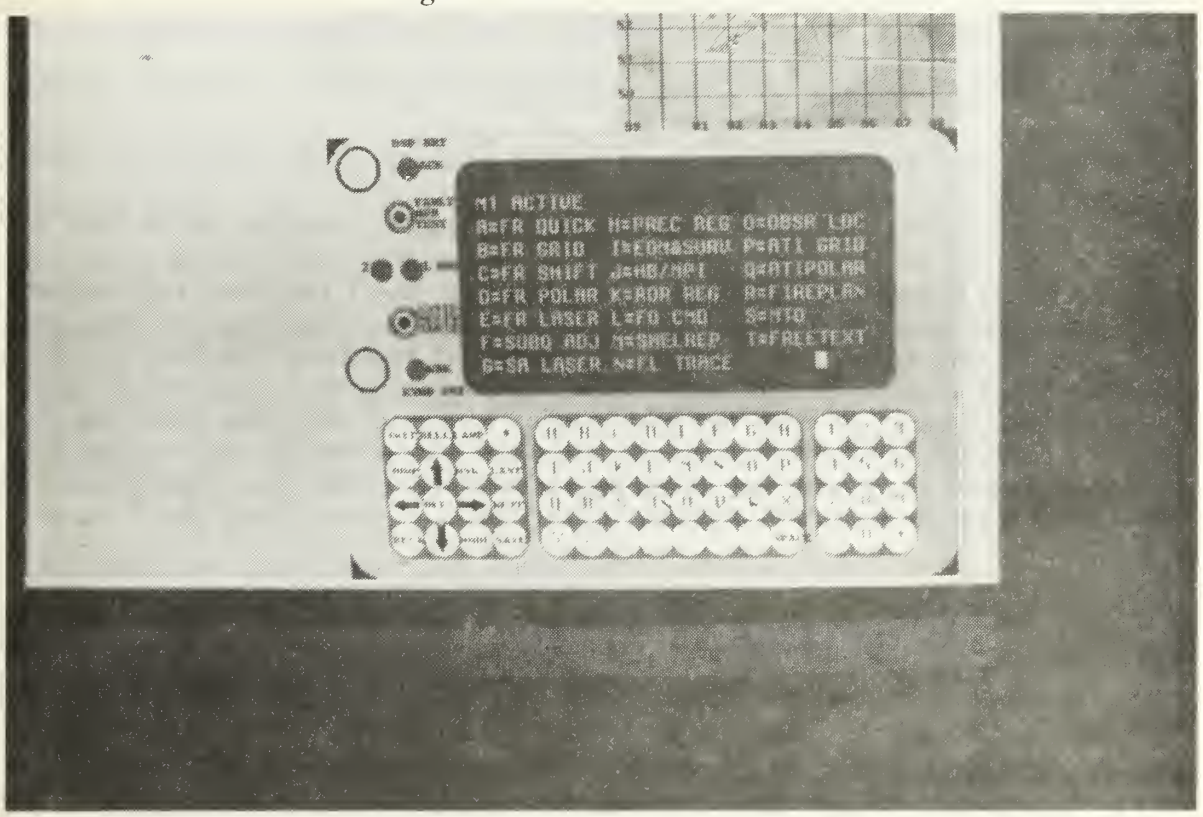


Figure 4.4 Message Types Menu

(FOST accepts any two letters for authentication), while the shell/fuze combination and fire control menus are option menus.

Both of the message sequences in our DMD have other menus which are unique based on their mission's particular purpose. Since the grid mission sequence produces the initial round, it has two menus that prompt the FO for the target grid easting and northing respectively. We chose not to implement the other menus that a real DMD would present during this sequence, such as target size, description and degree of protection. These menus are not vital in attacking targets in our simulation and we believe this does not degrade the training utility of FOST.

The menus unique to the subsequent adjust sequence provide the means for an FO to make corrections to the initial round's location. The corrections cause ensuing rounds to land on or closer to the target. First, the FO must indicate if he knows where the round landed by making the proper selection on the observed round (OBSN) menu. None of the selections in this menu have any effect on the mission and appear only for sequential correctness. Next, the FO can enter one or both lateral (LAT SHFT) and range (RG SHFT) information. FOST then makes the necessary calculations and adjusts the location of succeeding rounds in accordance with the FO's input. Again, as with the grid mission sequence, the subsequent adjustment sequence leaves out menus that would appear on a real DMD (i.e. target number and angle).

B. OTHER FIELD ARTILLERY CONSIDERATIONS

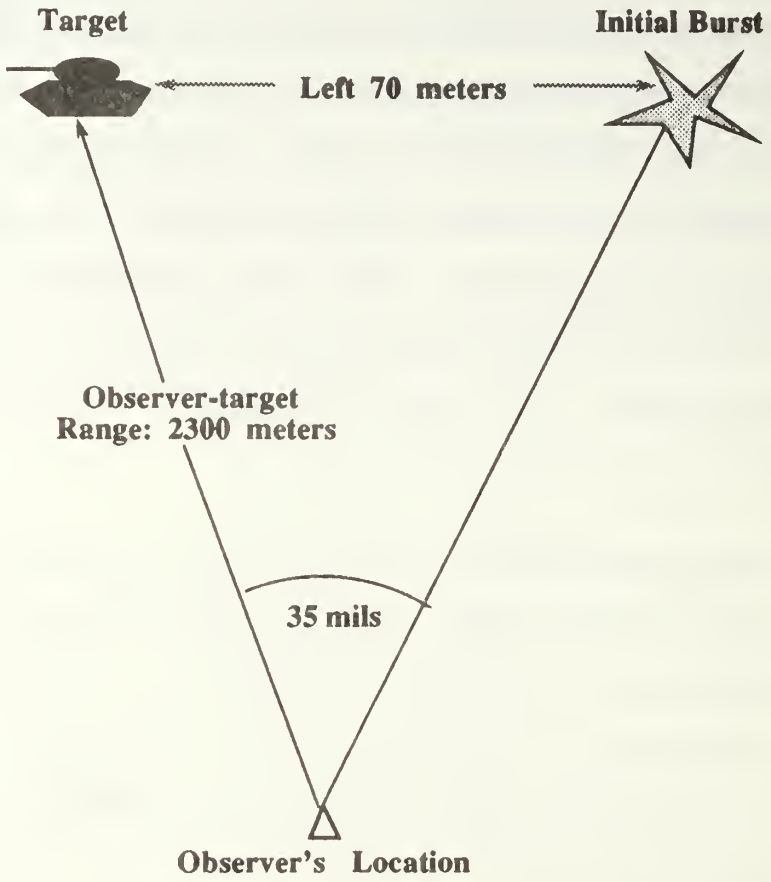
1. Binoculars

FOs use binoculars during fire missions for primarily two purposes: initial target location and subsequent adjustments. By using binoculars, FOs can more easily find distant and/or partially concealed targets. To make lateral corrections with more accuracy, the FO uses a geometric relation to determine the size of the shift in meters (see Figure 4.5). The calculation consists of simply multiplying the angular deviation, in mils, between the last round and the target by the observer's range to target., to the nearest kilometer. The result of this calculation is his lateral correction in meters.

Knowing the importance of binoculars from our experience as Field Artillerymen in the United States Army, we provide simulated binoculars in FOST to serve functions identical to real binoculars. To simplify use, we reserve the left mouse button as a toggle to turn the binoculars on and off. When toggled on, the binoculars appear as a reticle pattern (similar to US Army M-19 binoculars) on the 3D window. The scale on the reticle pattern is in mils, the standard Field Artillery unit of measurement for direction. When the reticle pattern appears on the screen, FOST narrows the field of view and in this way magnifies the terrain and objects, causing a zoom-in effect. Since we draw the binoculars using overdraw, the reticle pattern remains in place as the FO adjusts his viewing direction.

2. Projectile Effects On Targets

FOST has available four different types of artillery rounds: High Explosive/Point Detonating (HE/PD), High Explosive/Variable Time (HE/VT), High Explo-



Geometric relation: An angle of 1 mil subtends a distance of 1 meter at a range of 1000 meters.

The lateral shift calculation: Range in kilometers = 2;
Angular Deviation = 35 mils;
Lateral shift = 2 * 35 = Left 70 meters

Figure 4.5 Lateral Shift Formula

sive/Mechanical/Time (HE/TI) and Improved Conventional Munitions (ICM). In FOST, the three different terminal effects for the rounds - HE/VT and HE/TI are the same. We restrict the initial round of grid missions and subsequent adjustments to HE/PD, then use the type of round the FO specifies in the fire for effect. Since we simulate the Field Artillery's most current equipment, the fire for effect pattern is circular, the same pattern that the Battery Computer System (BCS)* generates (see Figure 4.6).

*BCS is a computer that provides firing data to the howitzers.

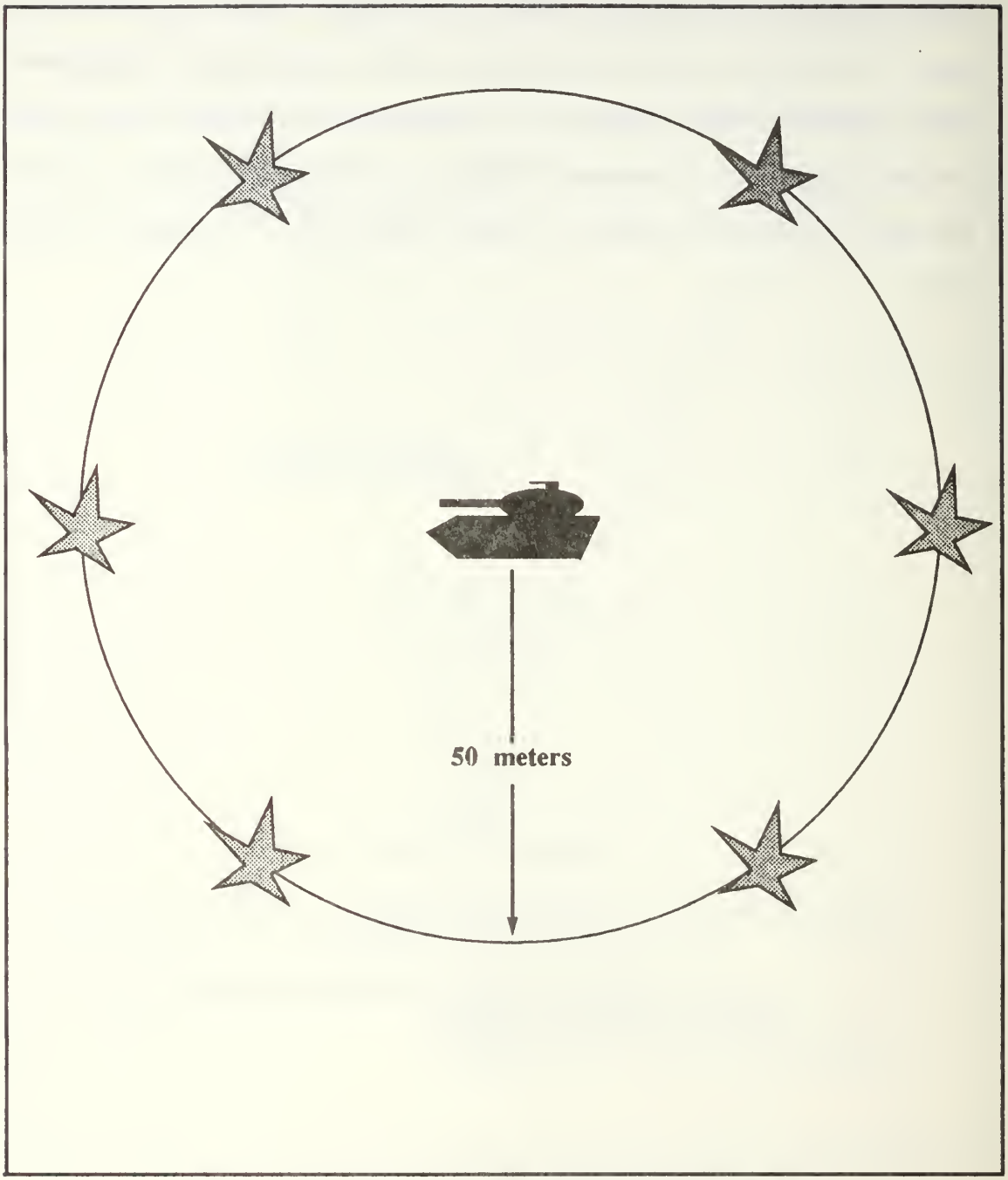


Figure 4.6 BCS Fire For Effect Pattern (6 gun battery)

V. COORDINATE ADJUSTMENT TECHNIQUES

A. COORDINATE TRANSFORMATIONS DURING SUBSEQUENT ADJUSTMENTS

1. Why Transformations Are Necessary

In FOST's simulated fire missions, as in real fire missions, it is necessary to know the grid location of the FO's adjusting point (where he intends to shoot). FOST, or the fire direction computers in a real situation, require this information in order to direct the rounds onto the target. If the FO is inaccurate in determining the initial location of the adjusting point, then he must make subsequent adjustments to hit the target.

When the FO makes his corrections, he does so from his perspective along the observer-target (OT) direction. Because the OT direction will, in general, not be due North, we must transform his correction coordinate system to the map coordinate system, which is oriented due North (see Figures 5.1 and 5.2). This transformation is a series of coordinate system translations and rotations that allow FOST to determine the map coordinate location of the next round.

2. How FOST Performs Coordinate Transformations

To illustrate what occurs when an FO conducts subsequent adjustments in FOST, refer to the example in Figure 5.3. In this situation, the FO should give a cor-

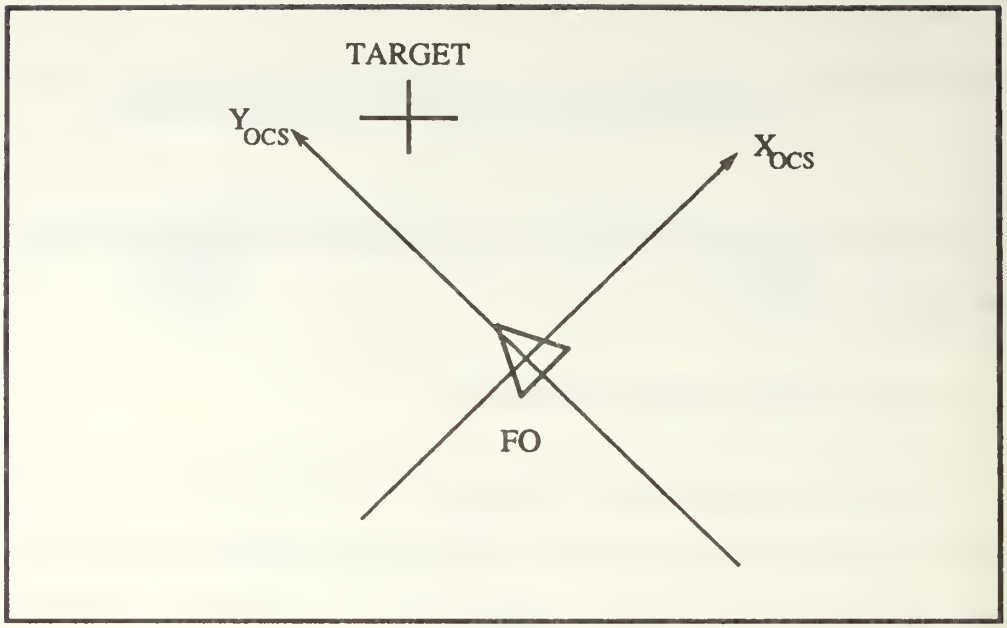


Figure 5.1 Observer Coordinate System (OCS)

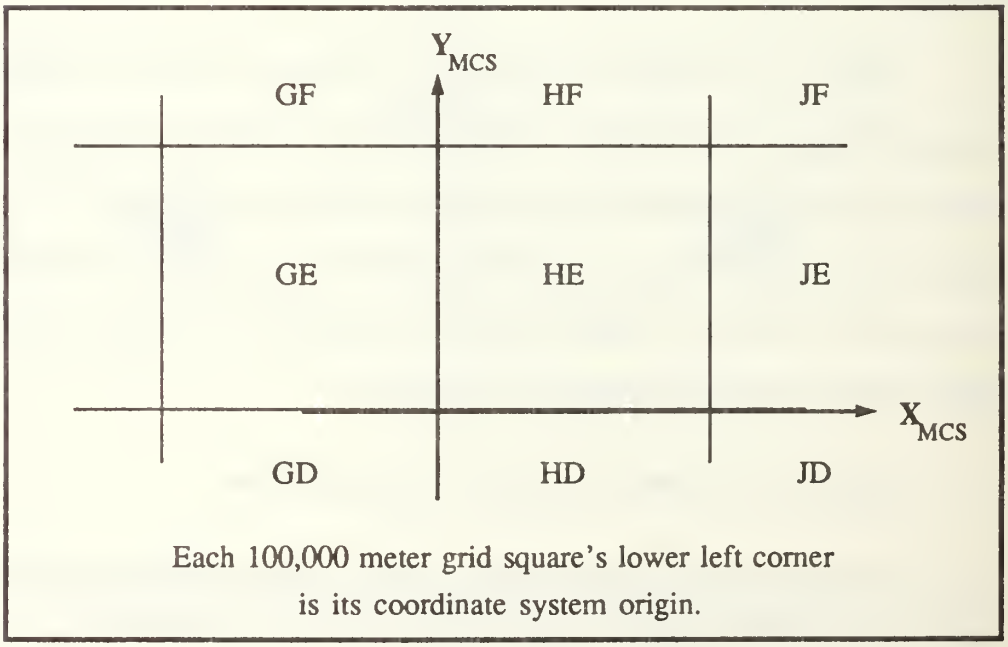


Figure 5.2 Map Coordinate System (MCS)

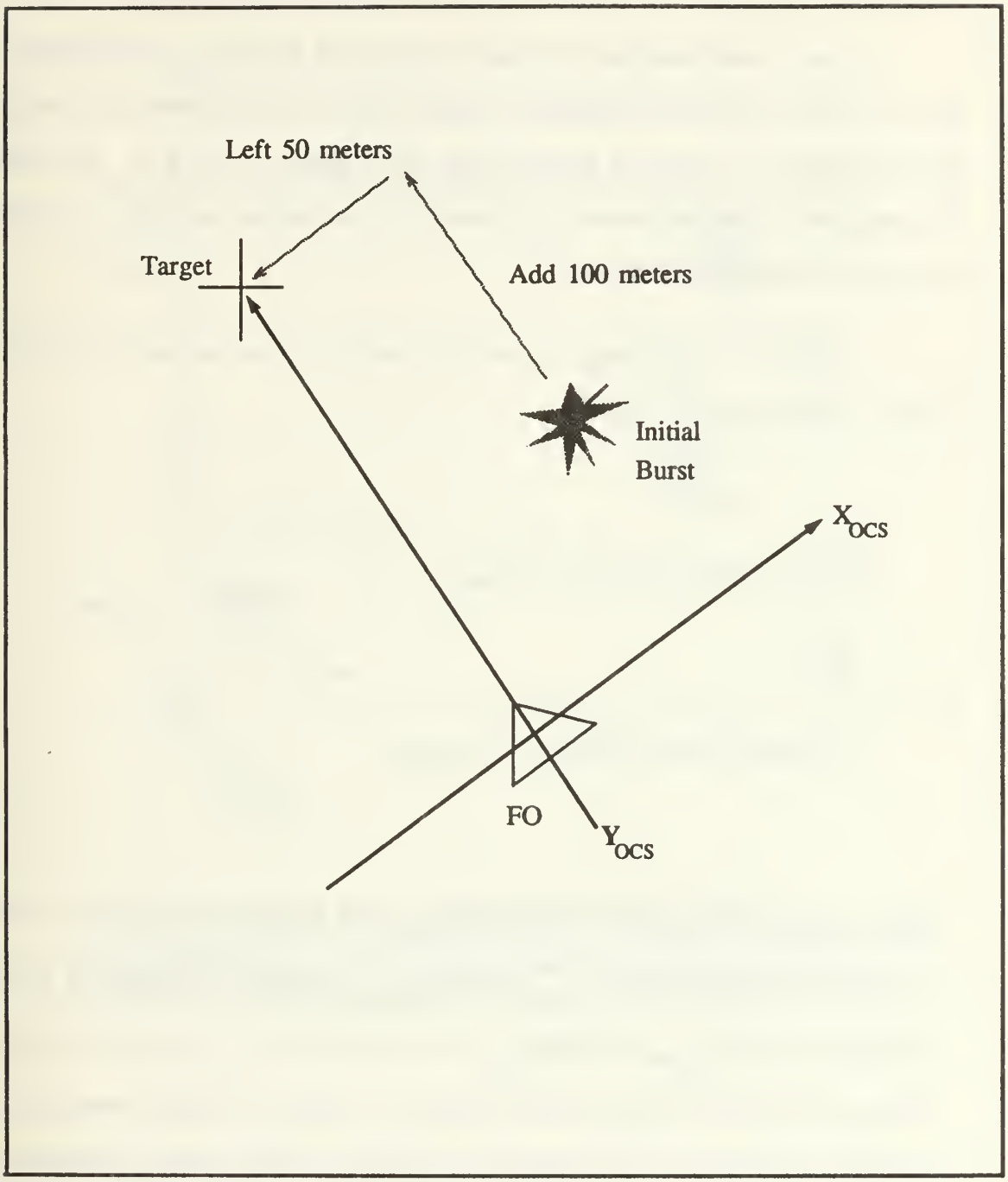


Figure 5.3 Subsequent Adjustment

rection of "Left 50, Add 100" in order to adjust the next round onto the target. Some calculations are necessary to make this adjustment.

First, a transformation is needed to convert the last round's grid location, in what we refer to as the Map Coordinate System (MCS), to a grid location in what we call the Observer's Coordinate System (OCS) (see Figures 5.1 and 5.2). The coordinate transform equations translate and rotate the two systems and give us the adjusting point's grid in the OCS.

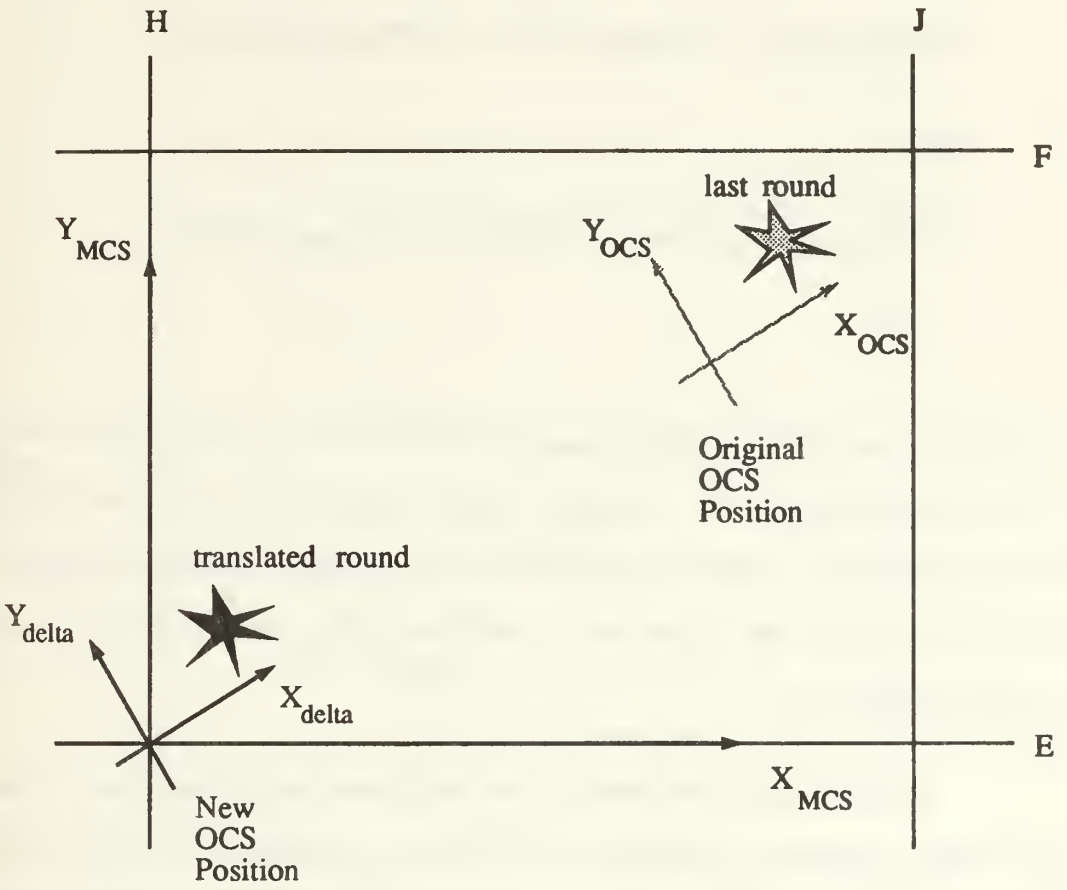
Translate the OCS origin to the MCS origin (it is just as correct to translate the MCS origin to the OCS origin):

$$East_{delta} = Round\_East_{MCS} - OP_{easting}$$

and

$$North_{delta} = Round\_North_{MCS} - OP_{northing},$$

where: $Round\_East_{MCS}$ and $Round\_North_{MCS}$ are the last observed round's MCS grid easting and northing; $OP_{easting}$ and $OP_{northing}$ comprise the observer's location in the MCS; and $East_{delta}$ and $North_{delta}$ are the differences, in meters, between the round and the observer's location (see Figure 5.4). Since the 100,000 meter grid line intersections represent MCS coordinate origins and the OP location represents the OCS origin, the $East_{delta}$ and $North_{delta}$ are the last round's coordinates in the translated coordinate system [Ref. 21].



The delta coordinate system represents the translated OCS. Finding the difference between the OP grid and the last observed round's grid (both in MCS) has the effect of transforming the last observed round's grid in MCS to a grid in the delta coordinate system.

Figure 5.4 OCS Translation To MCS Origin

Now rotate the MCS system to match the orientation of the translated OCS:

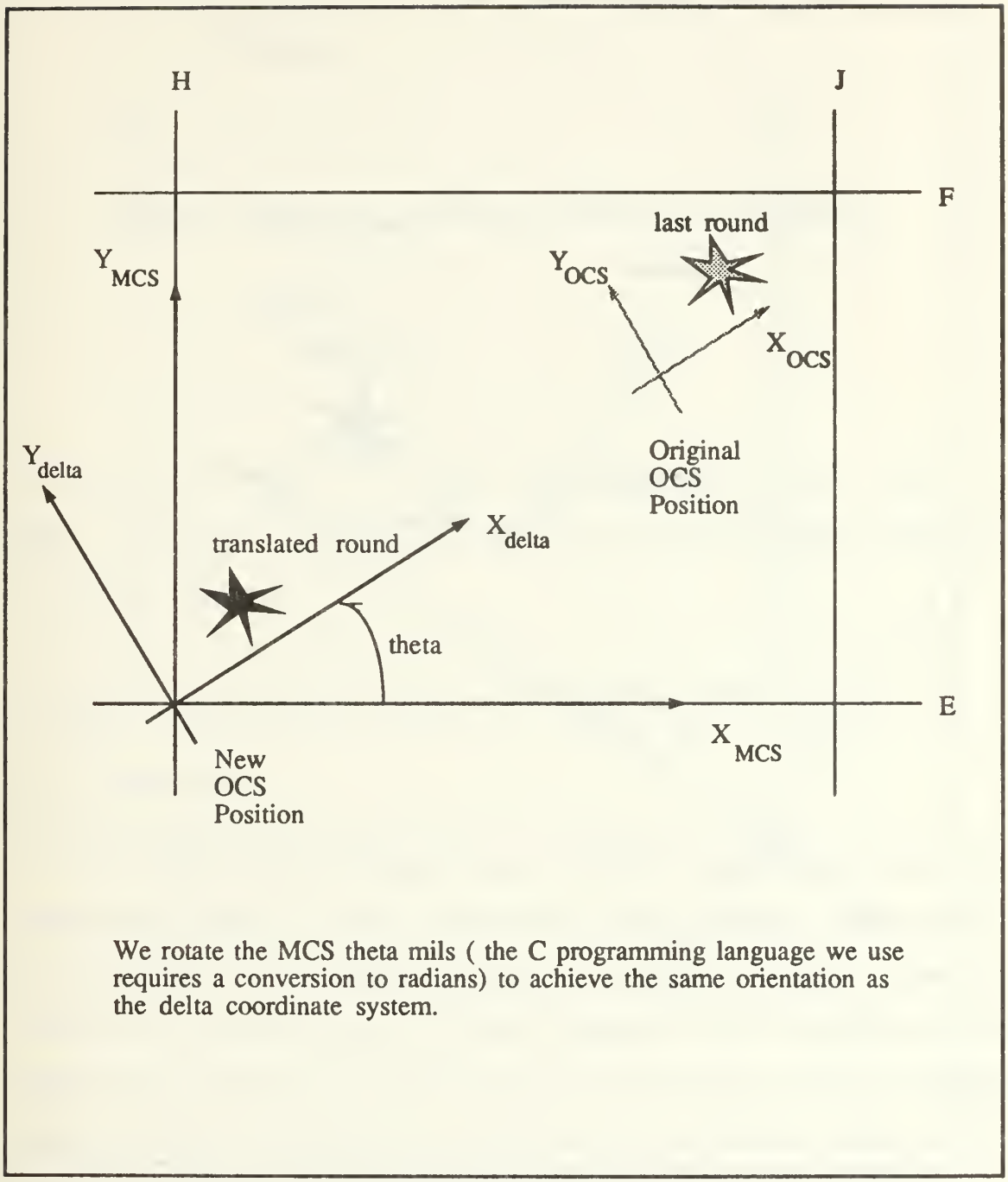
$$East_{MCS\_rotated} = East_{delta} \cos(theta) + North_{delta} \sin(theta)$$

and

$$North_{MCS\_rotated} = -East_{delta} \sin(theta) + North_{delta} \cos(theta)$$

$East_{MCS\_rotated}$ and $North_{MCS\_rotated}$ now hold the last round's grid easting and northing, respectively, in a coordinate system oriented the same as the OCS, but in MCS coordinates. Theta is the counter-clockwise angular difference between the x-axes (in this case, the x-axes are the MCS and the translated OCS eastings) (see Figure 5.5) [Ref. 21].

Once we reach this stage of the transformations, the observer's corrections are simple additions, for RIGHT and ADD, and subtractions, for LEFT and DROP (see Figure 5.6). After we apply the corrections, the result is the location of the adjusting point in the OCS. Now all that remains is a final coordinate transform back to the MCS. This time we do the rotation and translation in one set of equations. We rotate the MCS back to its original orientation and translate the OCS out to its original location in MCS coordinates [Ref. 20].



We rotate the MCS theta mils (the C programming language we use requires a conversion to radians) to achieve the same orientation as the delta coordinate system.

Figure 5.5 Rotate MCS To OCS Orientation

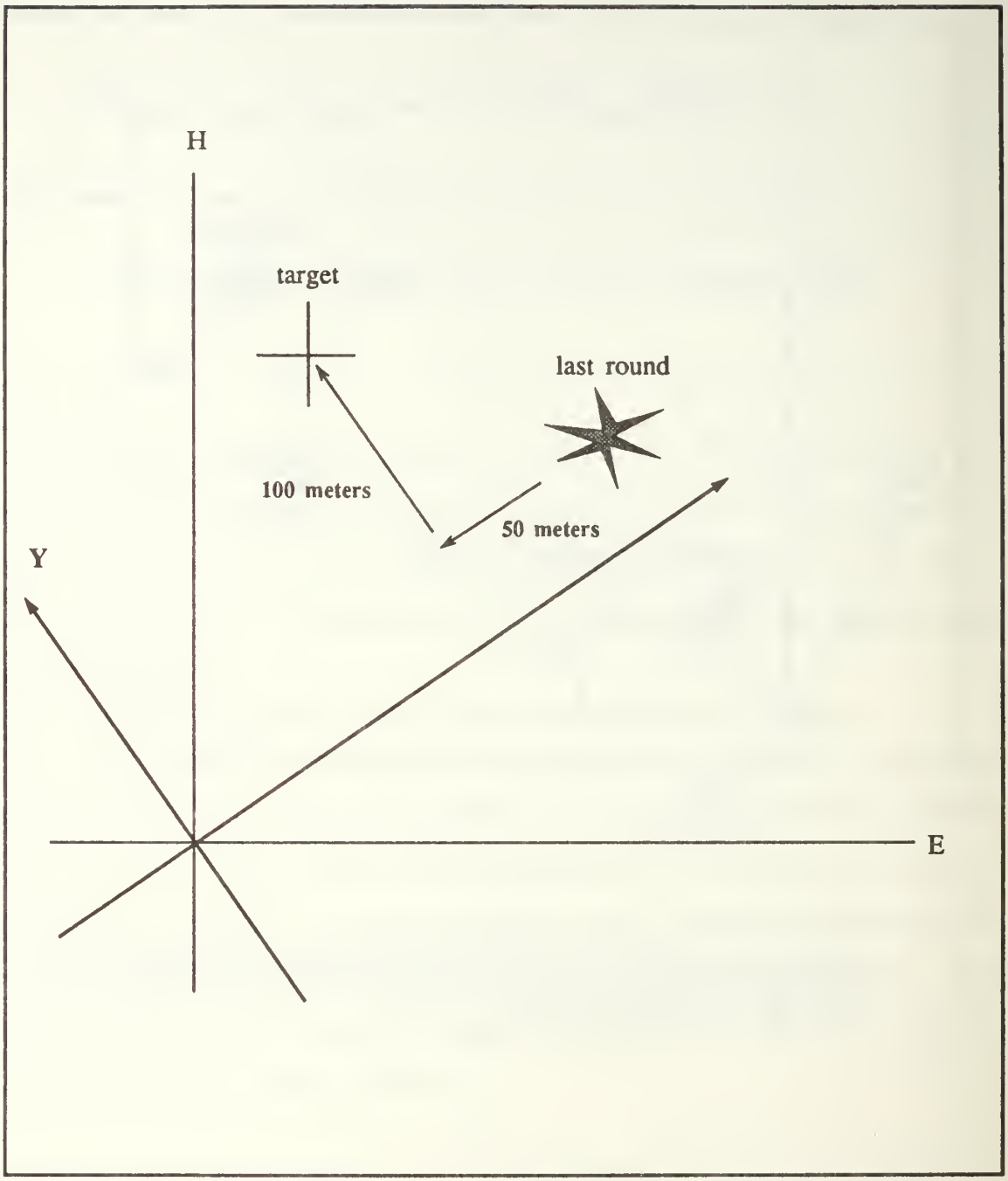


Figure 5.6 Perform Observer's Corrections

$$East_{MCS} = East_{MCS\_rotated} \cos(theta) - North_{MCS\_rotated} \sin(theta) + OP_{easting}$$

and

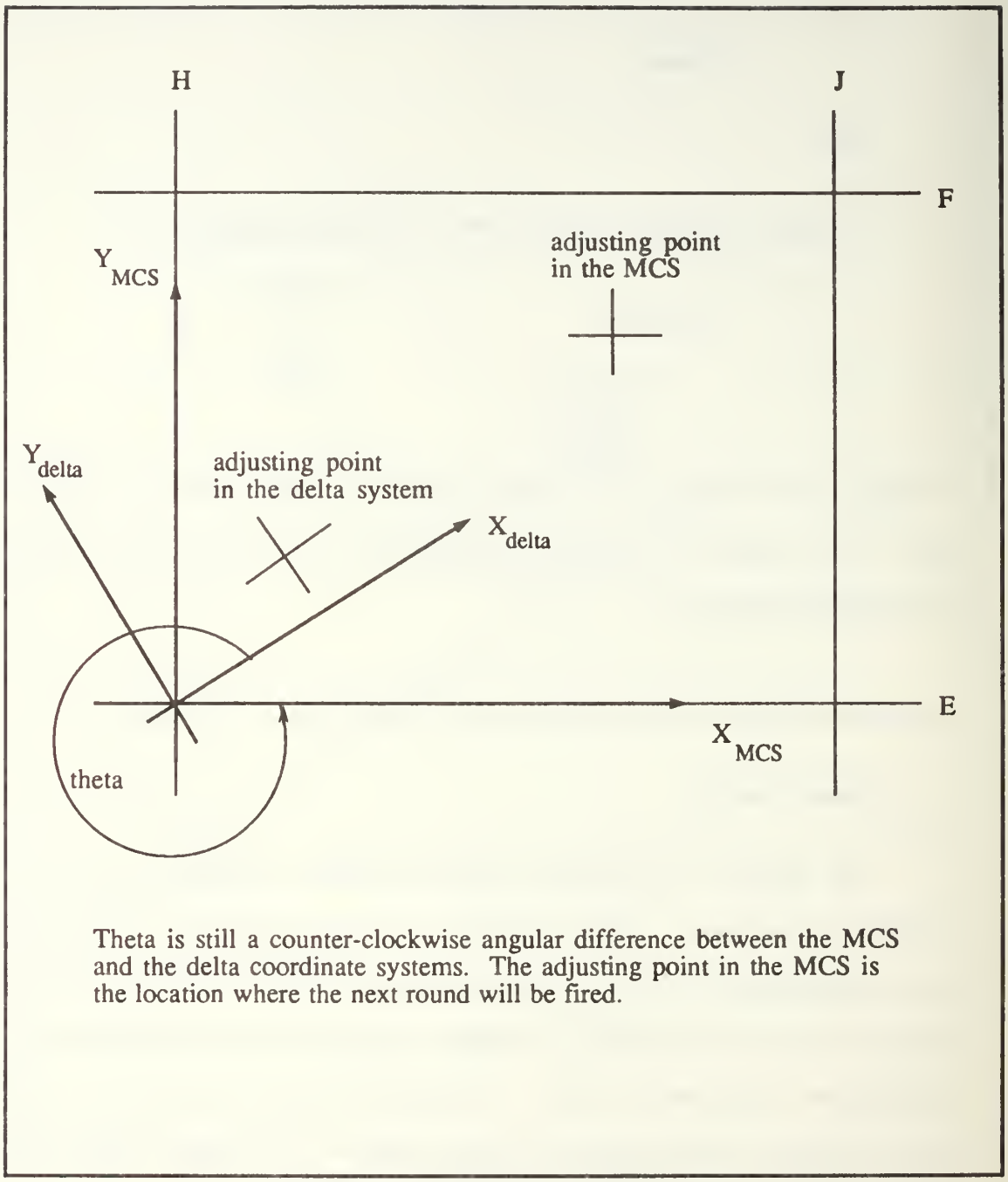
$$North_{MCS} = East_{MCS\_rotated} \sin(theta) + North_{MCS\_rotated} \cos(theta) + OP_{northing}$$

$East_{MCS}$ and $North_{MCS}$ now hold the adjusting point's easting and northing in MCS. This location is where the FO adjusted his fire and where FOST directs the next round (see Figure 5.7).

B. Special Considerations For Firing Across 100,000 Meter Grid Lines

1. Initial Rounds

The Military Grid Reference System (MGRS), subdivides grid zones into 100,000 meter squares. MGRS designates these squares with two letters and further subdivides by various degrees depending on the level of accuracy. Below the 100,000 meter grid square level, we cannot distinguish the grid coordinates of one 100,000 grid square from any other. The reason for this is that within each 100,000 meter grid square, the coordinates are purely numeric. For example, without the two letter designators, WD24683579 is the same as JN24683579.



Theta is still a counter-clockwise angular difference between the MCS and the delta coordinate systems. The adjusting point in the MCS is the location where the next round will be fired.

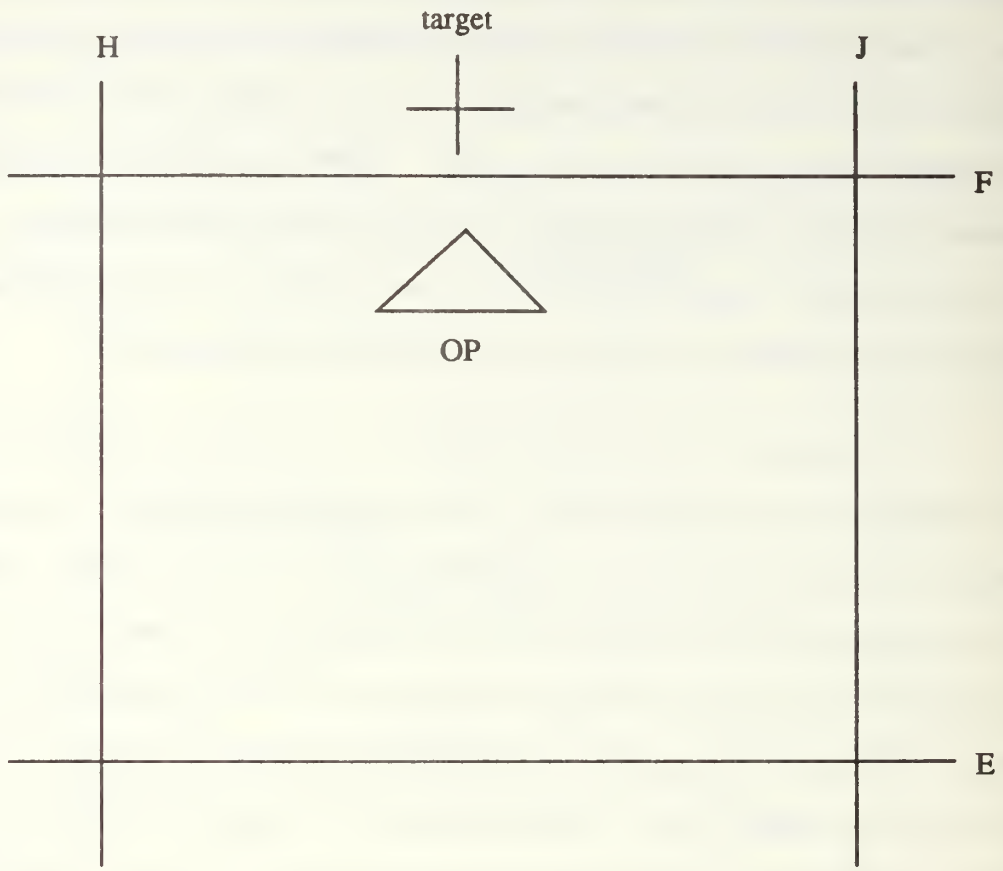
Figure 5.7 Final Transformation To MCS

The problem then, for FOST, is to determine when the FO is shooting across a 100,000 meter grid line. The DMD does not use the 100,000 meter grid zone designators, so the observation post (OP) and target coordinates can have only an eight digit accuracy. In the course of determining the round's time of flight (TOF), FOST figures the distance between the OP and the target. FOST uses the TOF to realistically simulate the length of time from firing the round to when the round detonates. Without a special check, firing across a 100,000 meter grid line as in Figure 5.8 results in an observer-target distance that is too long, which similarly affects the TOF.

To compensate for these type situations, we establish the restriction that the FO cannot fire more than 5000 meters. This is a reasonable restriction; in most cases targets are barely distinguishable at a 5000 meter range, let alone beyond. Additionally, most areas preclude visibility beyond 5000 meters due to either terrain contours or vegetation. We make this restriction so that if the FO and the target are in two different 100,000 meter grid squares, we effectively add 100,000 to the lesser coordinate and obtain the correct distance. Refering back to the example in Figure 5.8, FOST, with its 100,000 meter grid line check, correctly calculates the distance between the FO and the target (see Figures 5.9 and 5.10).

2. Subsequent Adjustments

FOST's problem with firing initial rounds across 100,000 meter grid lines remains just as true with adjustments after the initial round. In order to perform the necessary coordinate transformations, we make special allowances for crossing 100,000 meter grid lines. These allowances are simple; if the grid locations of the OP and the last observed round meet similar conditionals as in Figure 5.9 (replace target grid with last observed round grid), then we adjust the appropriate part of the grid



If the OP is located at grid HE 5000 9980 and the target is located at HF 5000 0020, then the actual distance is 400 meters. However, without a check for the 100,000 meter grid line, FOST would subtract the target location from the OP location and obtain a distance of 600 meters.

Figure 5.8 Initial Round Across A 100,000 Meter Grid Line

```
if (OP_easting > 9500) && (Target_easting < 5000)
   Target_easting = Target_easting + 10000;
if (OP_easting < 5000) && (Target_easting > 9500)
   OP_easting = OP_easting + 10000;
if (OP_northing > 9500) && (Target_northing < 5000)
  Target_northing = Target_northing + 10000;
if (OP_northing < 5000) && (Target_northing > 9500)
  OP_northing = OP_northing + 1000;
```

Figure 5.9 100,000 Meter Grid Line Conditionals

Since $OP_{northing} = 9980$ and $Target_{northing} = 0020$, the third conditional from Figure 5.9 applies. Then

$Target_{northing} = 0020 + 10000 = 10020.$

The distance Fost now calculates is:
10020 - 9980 = 40.

In meters this difference is 400 meters, the correct distance.

Figure 5.10 Example Calculation For Figure 5.8 Scenario

(i.e. easting or northing). We use a 10,000 meter factor, because our locations are accurate to ten meters (eight digits). This assures a correct distance between the OP and the last observed round for the initial translation just as it does with the initial round.

After FOST applies the FO's corrections to the last round and makes the last coordinate transformation, it makes another check for the 100,000 meter grid line case. If we applied the 10,000 meter factor, the final grid easting and/or northing for the next round will be greater than 10,000; we simply subtract 10,000 to obtain the correct grid coordinate.

VI. TERRAIN DRAWING IN FOST

Part of FOST's evolution is the change from the original terrain display technique, used by MPS, to the current method, which both MPS II and FOST use. While the current technique of using triangle mesh vertices generates more realistic looking terrain, it also contains much of MPS's terrain display algorithm. To fully understand the terrain process in FOST, it is important to look at both methods. In this chapter we give a broad-brush explanation of both algorithms. A complete explanation of both the terrain drawing technique and the lighting model is in [Refs. 1,8].

A. ORIGINAL TERRAIN DRAWING FROM MPS

1.Terrain Data Structures

After the user selects his ten by ten kilometer operating area, MPS creates two arrays which store information about the elevation and the terrain polygons. The *dted* array is a 101 x 101 array that stores the DTED file elevations for the area. MPS represents each 100 meter grid square using two triangles*, designated upper and lower, which creates a checkerboarding effect. The *gridcoord* array is an array that stores the X, Y, and Z values for each vertex of every terrain triangle [Ref 1].

---

* The reason for using triangles as the type of polygon is simple. Euclidean Geometry guarantees that triangles are planar, that guarantee does not hold for arbitrary four-pointed polygons. Using triangles keeps the system from drawing non-planar polygons, which would adversely affect system performance.

2.Terrain Display Algorithm

MPS displays only the terrain in the user's field of view. MPS begins the display procedure by computing the field of view. Based on the width and direction of the user's view angle, MPS determines the start and stop locations for the drawing routine in terms of their x and z grids (see Figure 6.1). In order to reduce degradation of the display performance, MPS uses a distance attenuation procedure. This procedure reduces the number of polygons the program displays by drawing the terrain using distance dependent resolution. The farther away the terrain is from the driven vehicle, the smaller the number of data points MPS uses (see Figure 6.2) [Ref 1].

Once MPS determines which polygons in the gridcoord array to draw, it begins the drawing process. MPS draws and colors the polygons using a coloring scheme based on the altitude (Y component) of the vertices. It also varies the color of alternate triangles to set up a checkerboarding appearance that gives the user a better view of the terrain contours [Ref 1].

3.Terrain Lighting In MPS

The MPS lighting model is the source of the terrain coloring and consequently the checkerboarding. The primary color scheme for the terrain forms a ramp of eight major colors each based on elevations of the given database. The colors range from lighter for lower elevations, to darker for higher elevations, which is similar to a standard map color scheme. There is also a secondary color scheme of eight minor colors (midway above the major colors). The lighting model uses these major and minor

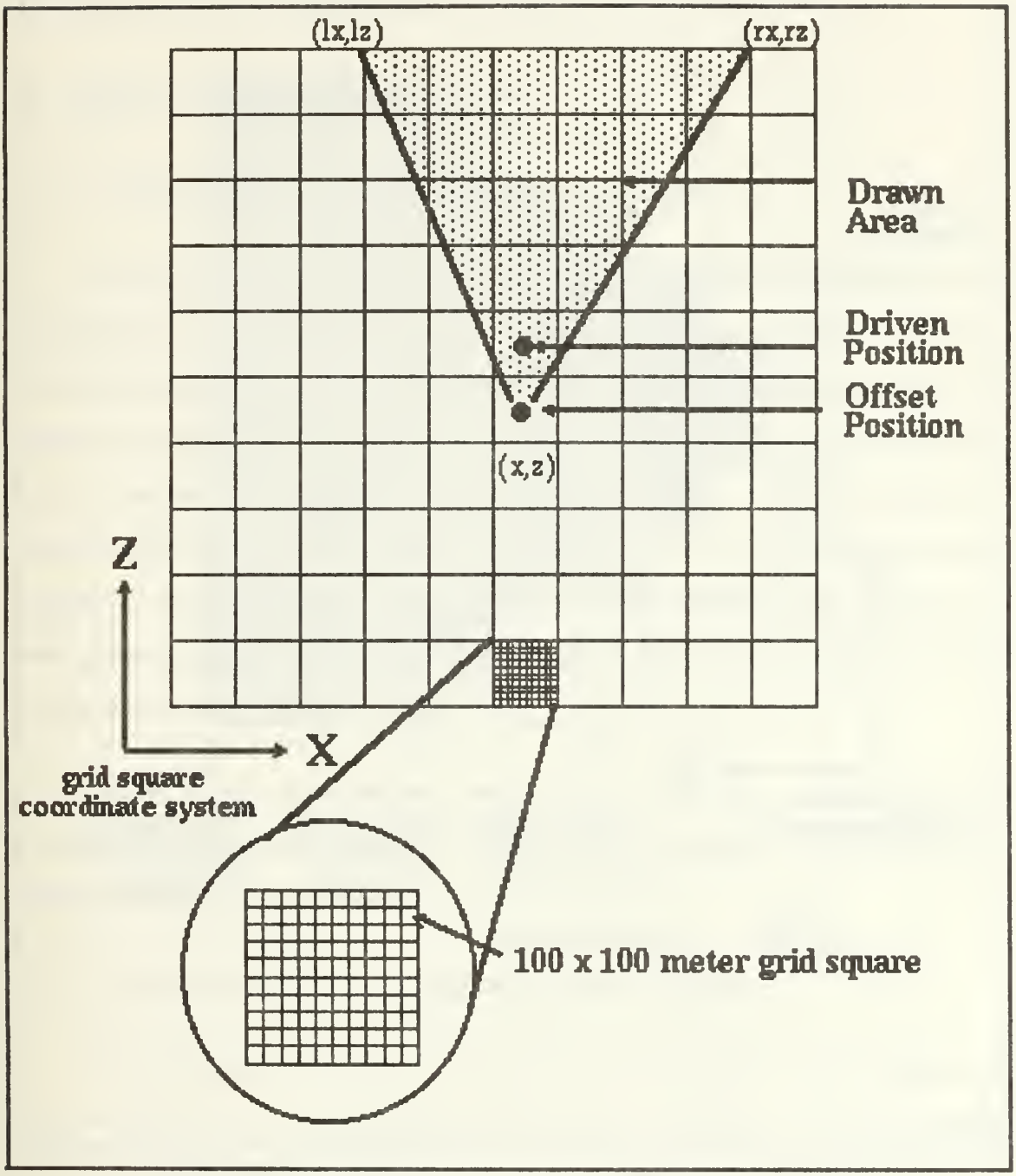


Figure 6.1 MPS and FOST Field-of-View Display

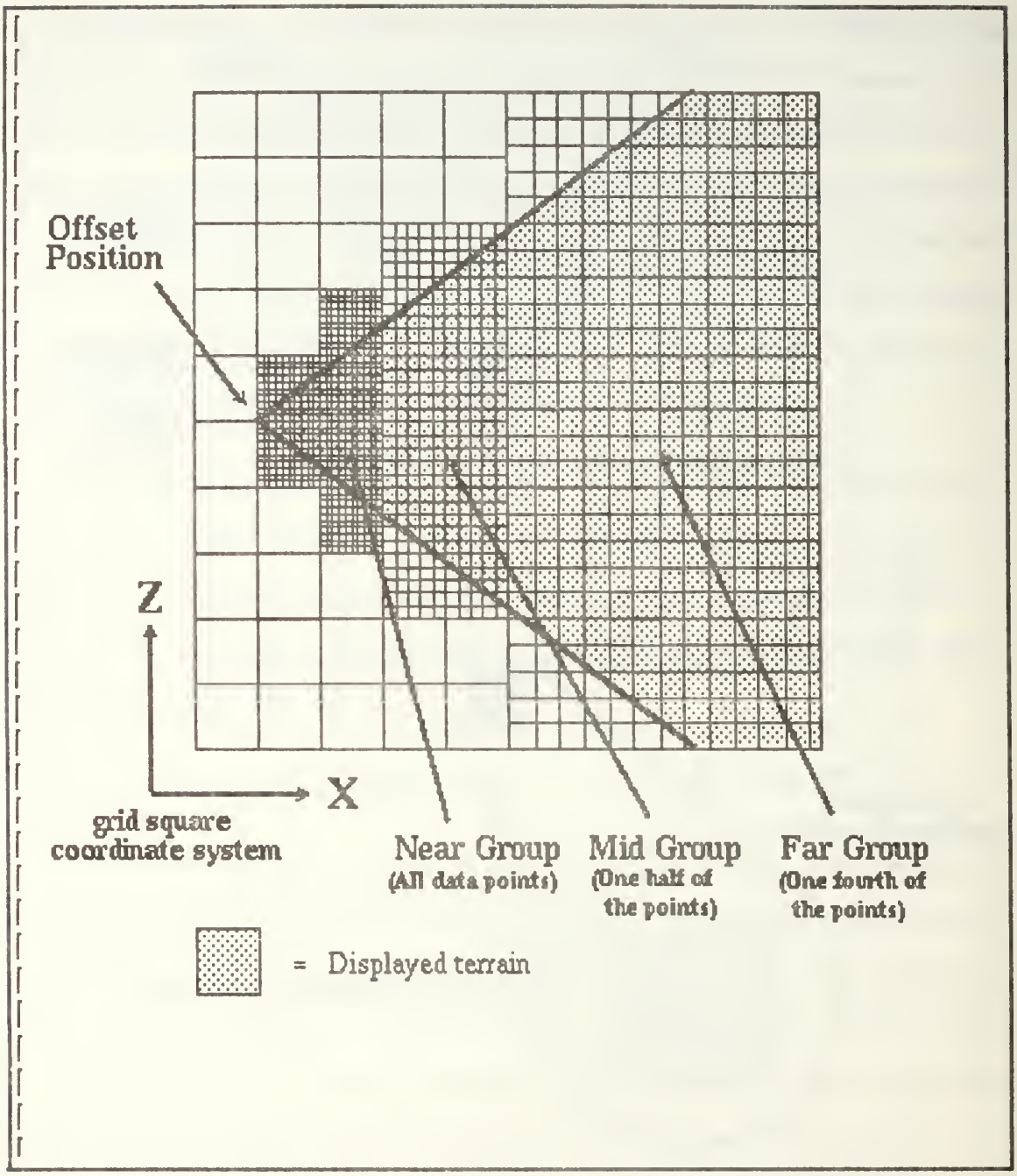


Figure 6.2 MPS and FOST Terrain Distance Attenuation

colors to create the checkerboarding effect. The even numbered triangles are colored with a major color and the odd numbered triangles are given a minor color.

B. TERRAIN DRAWING IN FOST

1. Adjustments To The Original Data Structures

Upon the start of our FOST simulation project, MPS performed efficiently and it seemed that we did not need to make adjustments to the data structures. On closer inspection of the *gridcoord* array we find that we are inefficiently storing data. Because this array stores the vertex coordinates of every terrain triangle, it contains a lot of redundancy. By using MPS's *gridcoord* structure, we store all of the coordinates multiple times; most of them six times (see Figure 6.3). We believed that adjustment to *gridcoord* would definitely improve storage efficiency and possibly improve program performance [Ref 8]. We also examined the *dted* structure but found no area for increased efficiency.

Before we dove headlong into changing this data structure, we had to examine what effect any adjustment would have on the drawing algorithm. We believe that coordinated efforts in programming achieve the best results.

2. Adjustments To The Original Drawing Algorithm

In examining the available options which the Iris provides in its graphics routines, we discovered a drawing primitive for a *mesh.* A mesh is a series of triangles that have common vertices [Refs 5, 6, 7]. If we implement mesh drawing of the terrain into the current MPS terrain drawing algorithm, we reduce the number of vertices we need to store and possibly improve program performance.

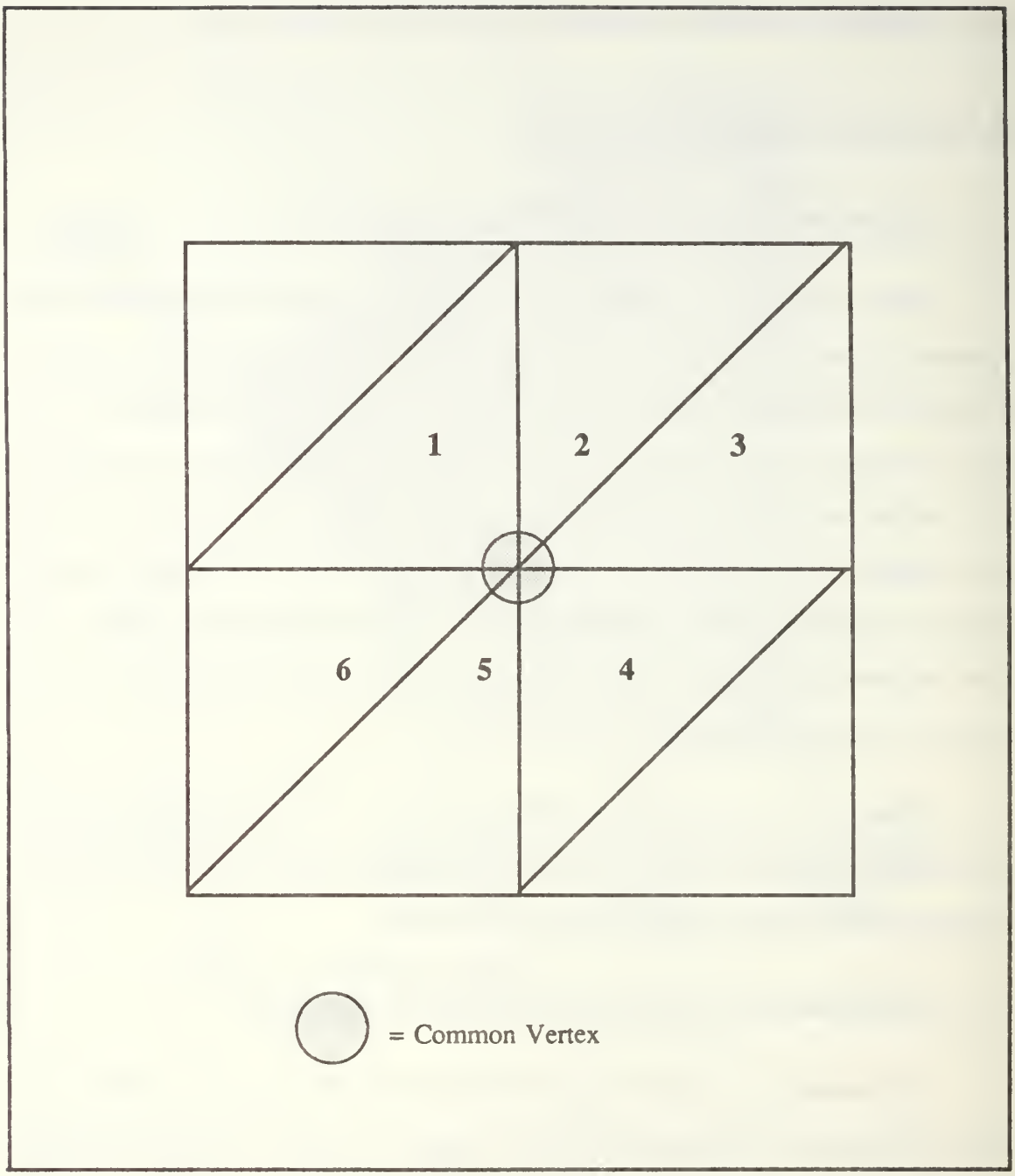


Figure 6.3 Common Vertex Problem

MPS II is the first descendent of MPS to use the mesh primitive [Ref 8]. With mesh drawing, the program sends a series of vertices to the function primitives. The primitive groups the first three vertices it receives into a polygon; thereafter, the method connects each new point to the previous two points, again forming a polygon. The result of this procedure is the elimination of redundant vertices. This procedure not only allows the use of a smaller terrain data structure, but due to its efficient operation, the developers of MPS II realize an increase in system performance of between three and five frames per second [Ref 8].

Following successful integration in MPS II, we modified FOST to use the triangle mesh procedure and increased our system performance between one and two frames per second. As in MPS II, we only needed to modify the *gridcoord* structure and use the Iris graphics routine functions: *bgntmesh()* and *endtmesh()* in our *drawterrain()* function. The modification did not affect either the distance attenuation procedure or the *dted* array.

3. Adjustments To Terrain Lighting

The lighting model MPS uses is another area that provides opportunities for increased efficiency. By reducing the overhead of the polygon normal array we increase our program efficiency. Instead of computing the true polygon normals as in MPS, MPS II uses an approximation of the true vertex normal. MPS II obtains this approximation by calculating the normal of one triangle, which shares a vertex with five other triangles, then extrapolating to the other five triangles. The difference between the terrain display with exact normal calculation and with normal approximation is negligible with the mesh technique [Ref 8]. This procedure also reduces the

size of the array that stores the vertex normals. Following this discovery in MPS II, we modified the normal calculation routine of FOST.

C. BENEFITS OF MESH DRAWING IN FOST

The triangle mesh technique coupled with this polygon normal calculation technique displays the terrain much more realistically than with checkerboarding (see Figures 6.4 and 6.5). We no longer need to distinguish adjacent triangles with major and minor color schemes. The lighting model shades the triangles based on their vertex normals in the same manner as the sun shades terrain. The additional benefit of increased program performance made the adoption of this technique from MPS II an easy decision.

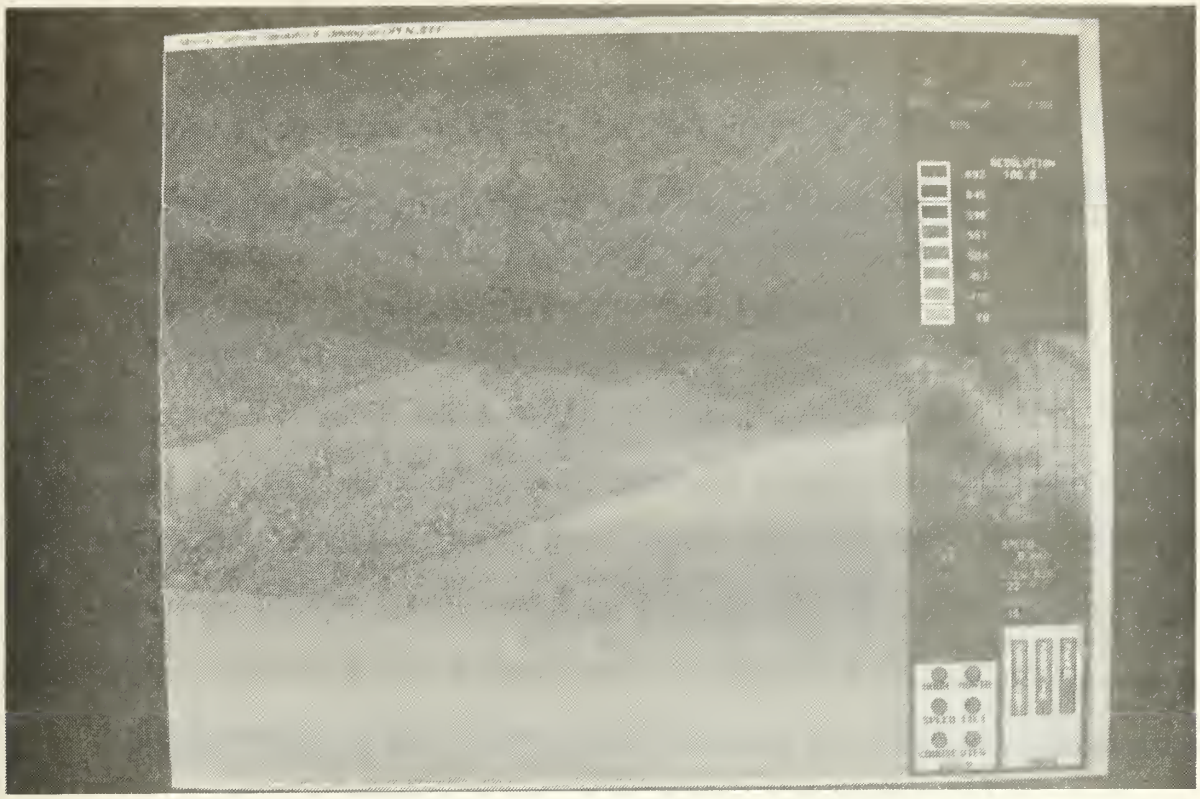


Figure 6.4 Terrain Display With Checkerboarding

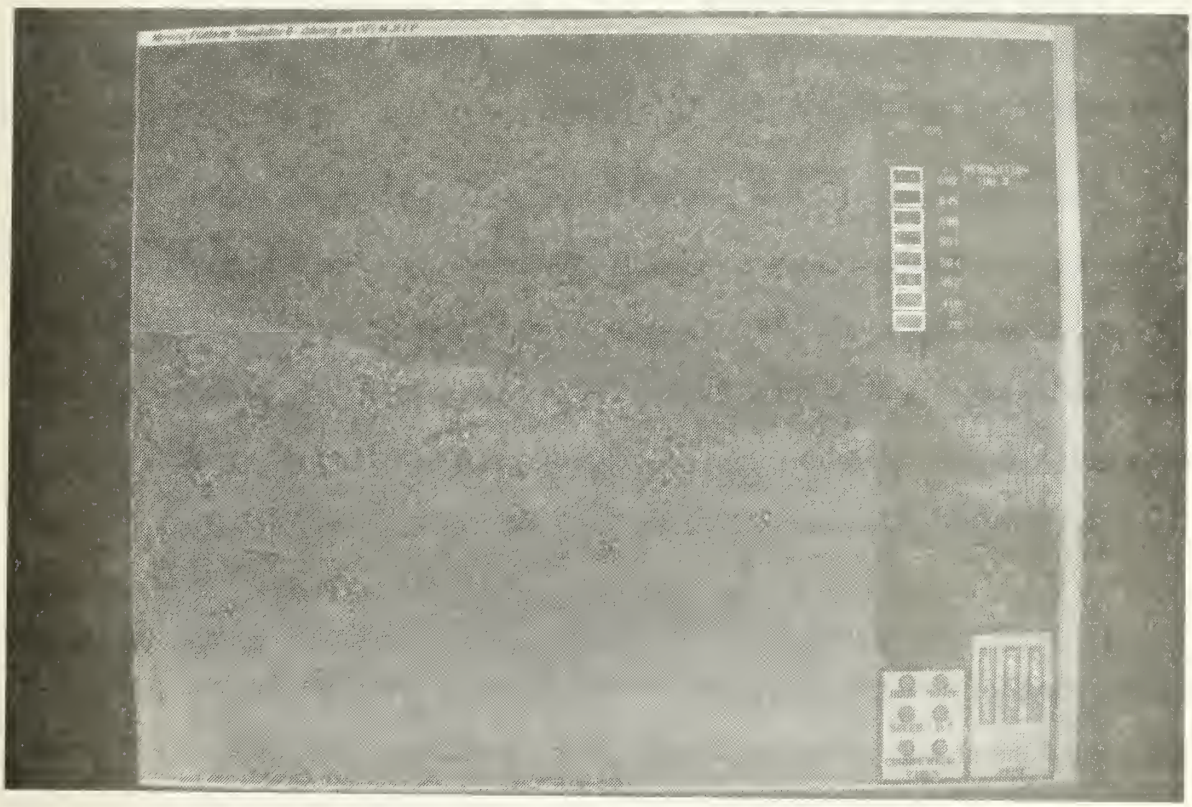


Figure 6.5 Terrain Display With Mesh

VII. CONVERSION OF GEOGRAPHIC TO MILITARY GRID REFERENCE SYSTEM (MGRS) COORDINATES

One of the most important features of FOST is the ability to load and display DMA Level I files. This allows FOs using FOST to "train" on calling for and adjusting indirect fire anywhere in the world. We reference these files by the location of their lower left hand corner, which DMA identifies with geographic coordinates - longitude and latitude. If we want the files to be useful to FOST, we must convert from the geographic location to the standard reference system of the U. S. Army [Ref 10].

A. A BRIEF HISTORY OF MGRS

1. What Is The MGRS And Why It Exists

There are numerous map projections in use throughout the world. The main reason for this great number is that different projections serve different purposes, some projections being better for a given application than others. Usually national interests determine which projection is the primary one which cartographers use in creating maps for a given nation. In the United States, the primary map projection is the Universal Transverse Mercator (UTM) [Refs 11, 12] .

The UTM projection originated during World War II when military requirements called for a world wide plane coordinate system based on the metric standard. The basic scheme was to flatten out and divide the surface of the Earth into rectangular areas six degrees in longitude by eight degrees in latitude. The end result is a 60

by 19 array of grid zones (see Figure 7.1). Each grid zone is designated by its number and letter as Figure 7.1 shows. The UTM grid system further subdivides each grid zone, commonly down to the 100,000th of a meter [Refs 11, 12].

In order to further identify locations in each of the UTM grid zones, the U. S. Army created the MGRS. The MGRS enhances the UTM system by subdividing each grid zone into 100,000 meter square areas and by using a two letter designation system (see Figure 7.2). The end result is an extremely accurate rectangular grid system to pin-point any location on the Earth. The MGRS is the standard system that the U. S. Army uses to locate positions on the Earth [Ref 10].

2. Why It Is Important To Convert

Because it is the U. S. Army standard, all locations that the Field Artillery uses are also in the MGRS. All Field Artillery fire direction computers send and receive location data using ten digit MGRS grids, which give a location accuracy to the nearest meter. The FO must transmit all locations to the computers using the MGRS to allow proper computation of gun firing data.

Since we designed FOST to be a training system for Field Artillery FOs, then it must also use the MGRS. In order to do that and use the DMA Level I files it must convert the geographic coordinates of the files to the MGRS. Proper and accurate conversion is important for FOST to correctly locate observers and targets.

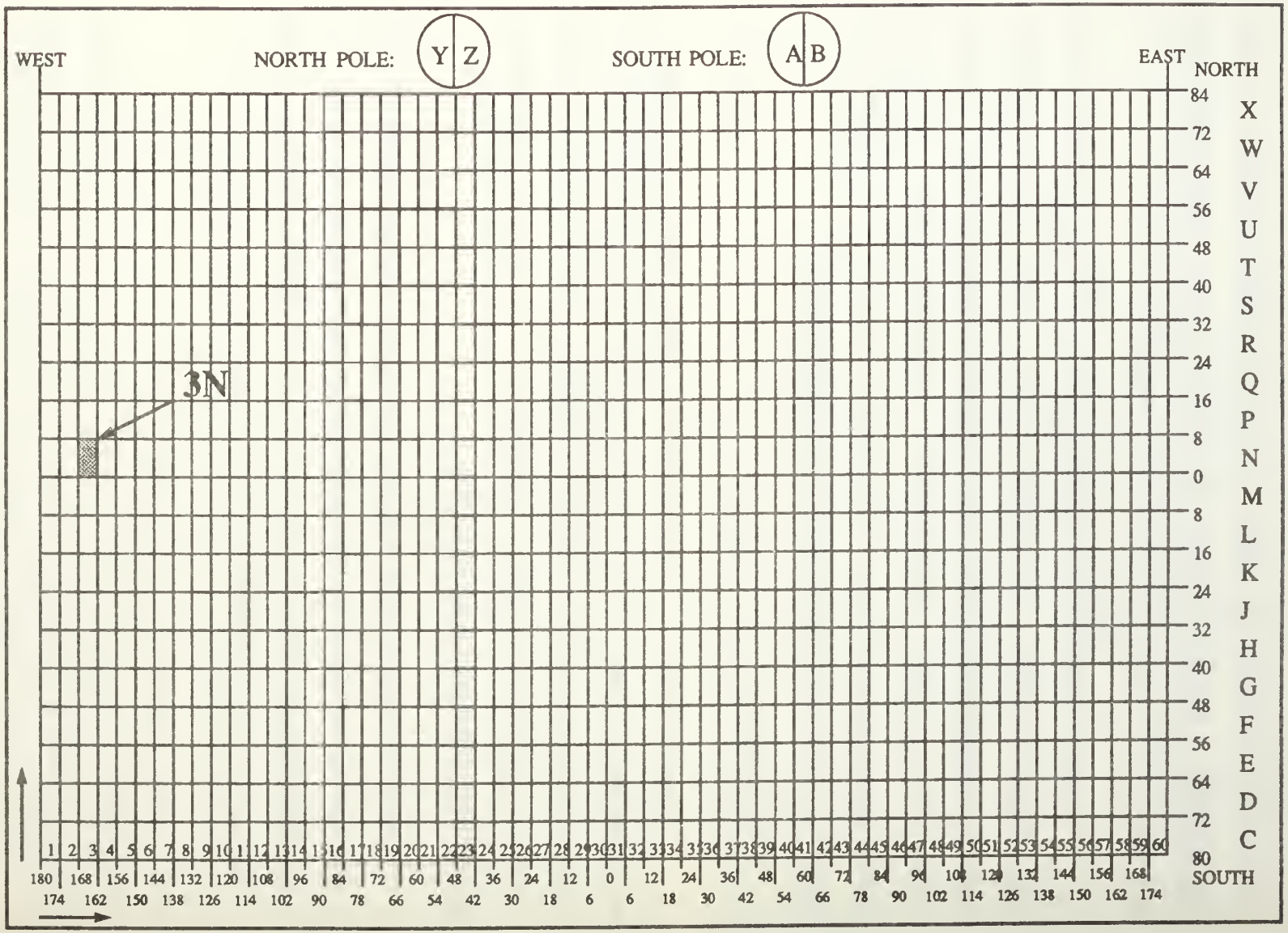


Figure 7.1 UTM Grid Zone Designations

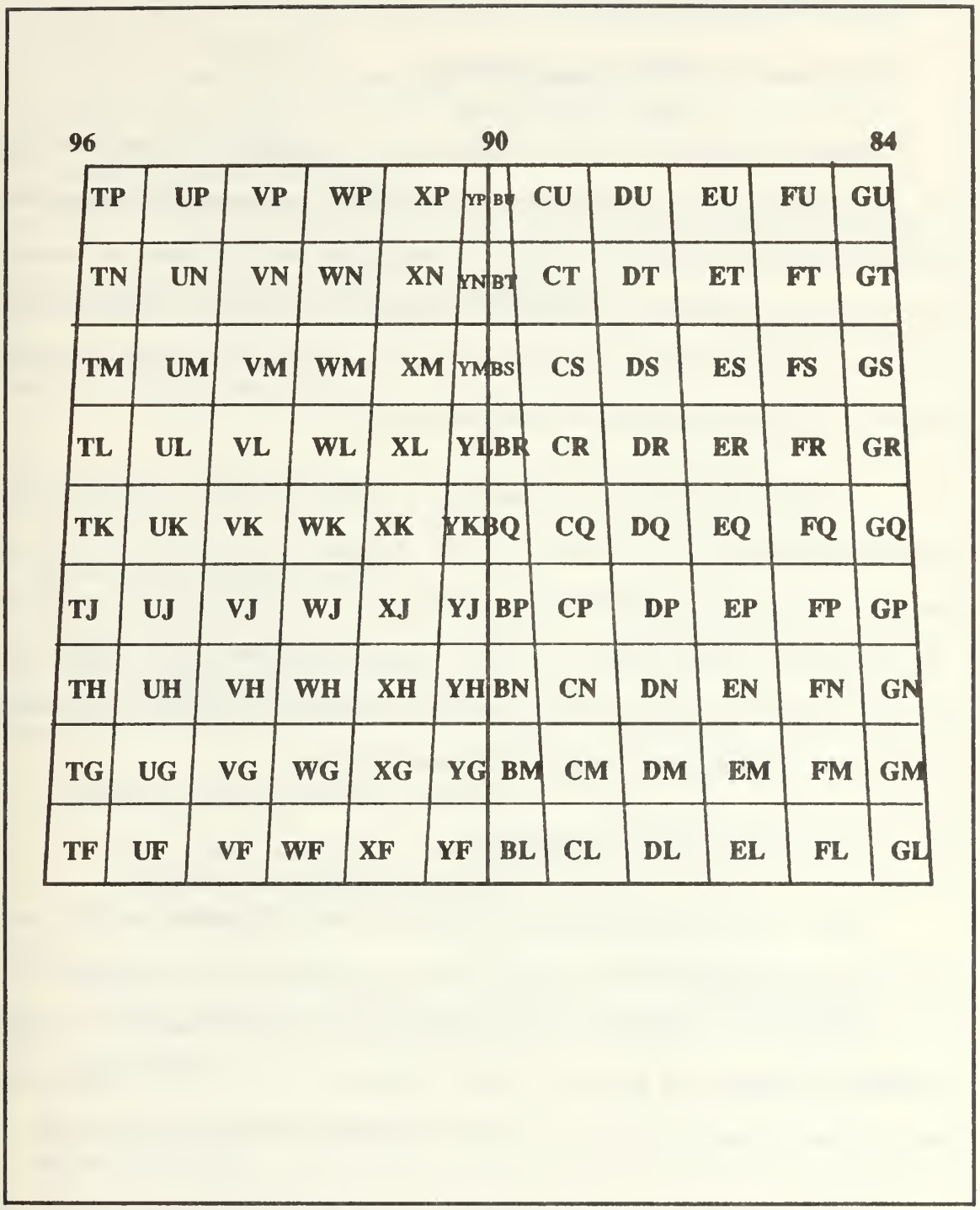


Figure 7.2 Example MGRS 100,000 Meter Square Designators

B. CONVERSION TECHNIQUE

1. Determination Of The Proper Spheroid

The initial step in the conversion process is to determine the spheroid in effect for the particular latitude and longitude. A spheroid is a mathematical figure which closely matches the surface of the Earth. Cartographers and surveyors use spheroids as a reference for geodetic surveys [Ref 11]. Because there exist several valid spheroids, each differing slightly with respect to the equatorial and polar radius, we must determine which one applies to our conversion location.

A spheroid determination amounts to a table lookup using the U. S. Army Technical Manual (TM) 5-241-series. If done manually, a user would enter the table using the latitude and longitude and find the spheroid. We modified a conversion of the information in these tables to an array of structures for use in FOST (see Figures 7.3and 7.4) [Ref 13]. The program uses this information to determine adjustments necessary to the UTM grid based on the determined spheroid.

2. Conversion To UTM Coordinates

Once we know the spheroid, the conversion from geographic to UTM coordinates is fairly straightforward. In FOST, we use an algorithm that appears in TM 5-237 as well as in a conversion program instructors at the United States Military Academy developed (see Figures 7.5 and 7.6) [Ref 13, 14]. We use several lookup tables, coded as arrays of structures, to solve this problem (see Figures 7.4 and 7.7).

The conversion routine, Geo2UTM, requires only the latitude, longitude, and spheroid to convert from geographic to UTM coordinates (see Figure 7.8). Geo2UTM models a Department of the Army Form 1932 which Army surveyors use for this con-

```
/*****************************************************************/
/* FUNCTION: spherelookup                                        */
/* This function looks up the correct spheriod for the given LAT,*/
/* LONG.  Returns true and the *sphere (as a side effect), returns*/
/* false if there is a problem.                                  */
/* The lookup array is stored in utm.h                           */
/* Calls function polylookup if the sphere is a special case.    */
/*****************************************************************/

Boolean spherelookup(lat, lon, conv_sphere, zoneoffset)

double lat, lon;
short *conv_sphere, *zoneoffset;

{
   int index;
   float tlat;

   if((lat <= 84.0) && (lat >= -80.0))
     {
      tlat = ((float)lat) * 60.0;
      index = swathIx[((int)lon) + 179];
      while(worldData[index].endlat <= tlat)
index = index + 1;

      if(worldData[index].spheroid != Special)
      {
          *conv_sphere = worldData[index].spheroid;
          *zoneoffset = worldData[index].offset;
      } /* end if Special */

      else
           polylookup(lat,lon,worldData[index].offset,*conv_sphere,*zoneoffset);

      return(TRUE);

    } /* end if lat between 84 and -80 */
   else return(FALSE);

} /* end spherelookup */
```

Figure 7.3 Spheroid Lookup Function

```
typedef struct  bounddata {short spheroid, endlat;
            short offset;
   } bounddata;

/* SEE World Spheroid Table */
/* Version 1.0  14 Oct 86 */

bounddata  worldData[2118] = {
   { International,  -3840,  0}, {      WGS72,   2760,  0},
   {    Clarke1866,   3720, 10}, {      WGS72,   5041,  0},
   { International,  -3840,  0}, {      WGS72,   2760,  0},
   {    Clarke1866,   3720, 10}, {      WGS72,   5041,  0},
   { International,  -3840,  0}, {      WGS72,   2760,  0},
   {    Clarke1866,   3720, 10}, {      WGS72,   5041,  0},
   { International,  -3840,  0}, {      WGS72,   2760,  0},
   {    Clarke1866,   3720, 10}, {      WGS72,   5041,  0},
   { International,  -3840,  0}, {      WGS72,   2760,  0},
   {    Clarke1866,   3720, 10}, {      WGS72,   5041,  0},
   { International,  -3840,  0}, {      WGS72,   2880,  0},
   {    Clarke1866,   3840, 10}, {      WGS72,   5041,  0},
   { International,  -3840,  0}, {      WGS72,   2880,  0},
   {    Clarke1866,   3840, 10}, {      WGS72,   5041,  0},
   { International,  -3840,  0}, {      WGS72,   2880,  0},
   {    Clarke1866,   3840, 10}, {      WGS72,   5041,  0},
   { International,  -3840,  0}, {      WGS72,   2880,  0},
   {    Clarke1866,   3840, 10}, {      WGS72,   5041,  0},
   { International,  -3840,  0}, {      WGS72,   2880,  0},
   {    Clarke1866,   3840, 10}, {      WGS72,   5041,  0},
   { International,  -3840,  0}, {      WGS72,   2880,  0},
   {    Clarke1866,   3840, 10}, {      WGS72,   5041,  0},
   { International,  -3840,  0}, {      WGS72,   3000,  0},
   {    Clarke1866,   5040, 10}, {      WGS72,   5041,  0},
   { International,  -3840,  0}, {      WGS72,   3000,  0},
   {    Clarke1866,   5040, 10}, {      WGS72,   5041,  0},
                           •
                           •
                           •  };
```

Figure 7.4 Data Array For Sperelookup Function

Conversion Algorithm (TM 5-237, Chap 13)

lat = latitude of point
lat0 = latitude of point in radians
lon = longitude of point
lat' = latitude of foot of perpendicular to the central meridian
delta lon = difference of longitude from the central meridian
dLat" = correction to latitude in seconds
dLon" = correction to longitude in seconds
dH = correction to elevation in meters
dX,dY,dZ = shifts between ellipsoid centers in meters
lon0 = longitude of the central meridian
a = semi-major axis of the spheroid
b = semi-minor axis of the spheroid
da = difference between semi-major axes of spheroids
df = difference between flattenings of spheroids

f = flattening or ellipticity = $(a-b) / a$

e^2 = eccentricity^2 = $(a^2 - b^2) / a^2$

$e'^2 = (a^2 - b^2) / b^2 = e^2 / (1 - e^2)$

$n = (a - b) / (a + b)$

k0 = scale factor at the central meridian = 0.9996
p = 0.0001 delta lon
q = 0.000001 (E - 500,000)

RN,v = radius of curvature in prime vertical

$$= a / ((1 - e^2 * \sin^2(lat))^{1/2}$$

$$RM = (a * (1 - e^2)) / ((1 - e^2 * \sin^2(lat))^{3/2}$$

S = meridional arc = $A' * lat0 - B' * \sin(2 * lat) + C' * \sin(4 * lat) - D' * \sin(6 * lat) + E' * \sin(8 * lat)$

$$A' = a * [1 - n + (5/4 * (n^2 - n^3)) + (81/64 * (n^4 - n^5) + ...]$$

$$B' = 3/2 * a * [n - n^2 + (7/8 * (n^3 - n^4)) + (55/64 * n^5 + ...]$$

$$C' = 15/16 * a * [n^2 - n^3 + (3/4 * (n^4 - n^5) + ...]$$

$$D' = 35/48 * a * [n^3 - n^4 + (11/16 * n^5) + ...]$$

$$E' = 315/51 * a * [n^4 - n^5 + ...]$$

Figure 7.5 Preliminary Algorithm Calculations

Conversion of Geographic Coordinates to UTM Coordinates

$I = S * k0$

$II = (v * \sin(lat) * \cos(lat) * \sin(1")) / 2 * k0 * 10^8$

$III = (\sin^4(1") * v * \sin(lat) * \cos^3(lat)) / 24 * (5 - \tan^2(lat) + 9 * e' * (\cos^2(lat))^2 + 4 * e' * (\cos^4(lat))^4) * k0 * 10^{16}$

$IV = v * \cos(lat) * \sin(1") * k0 * 10^4$

$V = (\sin^3(1") * v * \cos^3(lat)/6) * (1 - \tan^2(lat) + e'^2 * \cos^2(lat)) * k0 * 10^{12}$

$A6 = p6 * (\sin^6(1") * v * \sin(lat) * \cos^5(lat) / 720) * (61 - 58 * \tan^2(lat) + \tan^4(lat) + 270 * e'^2 * \cos^2(lat) - 330 * e'^2 * \sin^2(lat)) * k0 * 10^{24}$

$B5 = p5 * (\sin^5(1") * v * \cos^5(lat)/720) * (5 - 18 * \tan^2(lat) + \tan^4(lat) + 14 * e'^2 * \cos^2(lat) - 58 * e'^2 * \sin^2(lat)) * k0 * 10^{20}$

$N = I + II * p^2 + III * p^4 + A6$

$E' = IV * p + V * p^3 + B5$

$E = 500{,}000 +/- E'$

Figure 7.6 UTM Conversion Algorithm

```
typedef struct  spheroiddata {  char Name[16];
            double Ap, Bp, Cp, Dp, Ep, a, f, e2, ep2;
  } spheroiddata;

spheroiddata spheroids[12] = {
 { "NULL",
    0.0,   0.0,   0.0,
    0.0,   0.0,   0.0,
    0.0,   0.0,   0.0},
 { "WGS 72",
    6.3674472486e6,   1.6038354332e4,   1.6832294336e1,
    2.1983782605e-2,   3.1269345895e-4,   6.3781350000e6,
     3.3527794542e-3,   6.6943177770e-3,   6.7394336877e-3},
 { "INTERNATIONAL",
    6.3676545001e6,   1.6107034678e4,   1.6976210952e1,
    2.2265965136e-2,   3.1805304041e-4,   6.3783880000e6,
     3.3670033670e-3,   6.7226700221e-3,   6.7681701970e-3},
 { "GRS 67 {Aust.}",
    6.3674718485e6,   1.6038954946e4,   1.6833490018e1,
    2.1986082604e-2,   3.1273667664e-4,   6.3781600000e6,
     3.3528918692e-3,   6.6945418532e-3,   6.7396607945e-3},
 { "Clarke 1866",
    6.3673996892e6,   1.6216944155e4,   1.7209371179e1,
    2.2726710261e-2,   3.2686272983e-4,   6.3782064000e6,
     3.3900753037e-3,   6.7686579969e-3,   6.8147849455e-3},
 { "Clarke 1880",
    6.3673866440e6,   1.6300700648e4,   1.7387630250e1,
    2.3080760496e-2,   3.3366993682e-4,   6.3782491450e6,
     3.4075613787e-3,   6.8035112823e-3,   6.8501161246e-3},
 { "EVEREST",
    6.3666802917e6,   1.5900693381e4,   1.6546576187e1,
    2.1427712637e-2,   3.0220438981e-4,   6.3772763452e6,
     3.3244492967e-3,   6.6378466276e-3,   6.6822020600e-3},
 { "Modified Everest",
    6.3667079634e6,   1.5900762491e4,   1.6546648106e1,
    2.1427805773e-2,   3.0220570337e-4,   6.3773040630e6,
     3.3244492967e-3,   6.6378466280e-3,   6.6822020604e-3},
                      •
                      •
                      •  };
```

Figure 7.7 Data Structure For UTM Calculations

version process. FOST finds the conversion location in latitude and longitude from the DMA file header when it reads in the file. The conversion returns the values of the UTM northing and easting and the offset of the longitude from the Central Meridian [Ref 13, 14].

3. Conversion of UTM to MGRS

The final step is to refine the UTM to the MGRS, and we use the function UTM2MGRS to accomplish this (see Figure 7.10). Again the process is straightforward; we already have the location in the UTM grid, the MGRS is merely a different format. Since the UTM coordinate is a pure numerical expression, we first determine the UTM grid zone alphanumeric designation using the latitude and the Central Meridian offset (see Figure 7.1). Next we find the 100,000 meter square identification with simple modulo arithmetic operations. For the first letter we operate on the Central Meridian offset and the UTM easting. We use the latitude offset and the UTM northing to find the second letter [Ref 15]. To complete the MGRS grid, we use the UTM easting and northing, taking the first five digits to the left of the decimal place to form our MGRS easting and northing. The conversion routine returns the completed MGRS to FOST for use throughout the program (see Figure 7.11).

```
Geo2UTM(lat, lon, Northing, Easting, CM, conv_sphere)

double lat,lon, *Northing,*Easting,*CM;
short conv_sphere;
{
  /* Variables to model the surveyor's DA Form 1932. */
  double DA1932_I,DA1932_II,DA1932_III,DA1932_IV,DA1932_V,DA1932_A6,DA1932_B5;
  Boolean South;

  *CM = Geo2CM(lat, lon, conv_sphere);       South = (lat < 0);

  lat = fabs(lat / RadToDeg);

  /* Preliminary calculations. */
  Rho = fabs(lon - *CM) * 0.36;        Sinlat = sin(lat);        Coslat = cos(lat);
  Nu = spheroids[conv_sphere].a / (sqrt(1 - (spheroids[conv_sphere].e2 * Sinlat * Sinlat)));
  Sin2lat = pow(sin(lat),2.0);          Cos2lat = pow(cos(lat),2.0);
  Tan2lat = pow(Sinlat/Coslat,2.0);     Tan4lat = Tan2lat * Tan2lat;
  ep2Cos2lat = spheroids[conv_sphere].ep2 * Cos2lat;
  ep2Sin2lat = spheroids[conv_sphere].ep2 * Sin2lat;

  /* DA 1932 calculations */
 DA1932_I    = k0 * (spheroids[conv_sphere].Ap * lat - spheroids[conv_sphere].Bp * sin(2 * lat) +
spheoids[conv_sphere].Cp * sin(4 * lat) - spheroids[conv_sphere].Dp * sin(6 * lat) + sphe-
roids[conv_sphere].Ep * sin(8 * lat));

  DA1932_II = (factor1(2.0,1.0,1.0,0.0) / 2) * k0 * 1e8;

 DA1932_III = (factor(4.0,1.0,3.0,0.0) / 24) * (factor2(5.0,1.0,0.0,9.0,0.0) + (4 * pow( pow( sphe-
roids[conv_sphere].ep2,Coslat),2.0)) * k0) * 1e16;

  DA1932_A6 = (factor1(6.0,1.0,5.0,6.0) / 720) * factor2(61.0,58.0,1.0,270.0,330.0) * 1e24;

  DA1932_IV = factor1(1.0,0.0,1.0,0.0) * k0 * 1e4;

  DA1932_V  = (factor1(3.0,0.0,3.0,0.0) / 6) * factor2(1.0,1.0,0.0,1.0,0.0) * 1e12;

  DA1932_B5 = (factor1(5.0,0.0,5.0,5.0) / 120) * factor2(5.0,18.0,1.0,14.0,58.0) * 1e20;

  *Northing  = DA1932_I + DA1932_II * Rho * Rho + DA1932_III * pow(Rho,4.0) + DA1932_A6;

  *Easting   = DA1932_IV * Rho + DA1932_V * pow(Rho,3.0) + DA1932_B5;

  /* Adjust for Southern latitudes and Western longitudes */
  if(South) *Northing = 1E7- *Northing;

  if((lon - *CM) > 0) *Easting = *Easting + 500000.0;
  else *Easting = 500000.0 - *Easting;

} /* end Geo2UTM */
```

Figure 7.8 Geo2UTM Function

```
UTM2MGRS(Northing, Easting, CM, lat, lon, toffset, MGRS)

double Northing, Easting, CM, lat, lon;
short toffset;
char *MGRS;
{
 float   tN;
 int   j;
char   MGRS1[3], MGRS2[2], MGRS3[2], MGRS4[2], NStr[6], EStr[6];

   /* set the first character to the NULL string. */
   MGRS1[0] = '\0'; MGRS2[0] = '\0'; MGRS3[0] = '\0'; MGRS4[0] = '\0'; NStr[0] = '\0'; EStr[0] = '\0';
   MGRS[0] = '\0';

    /* Determine the letter of the MGRS grid zone designator.  A straightforward process.  The first three digits
of the  MGRS are the UTM grid zone.  The letter comes    from the latitude  and the number comes from the lon-
gitude.  See Maps for America  page 240 for an illustration.                                                  */

   sprintf(MGRS2,"%c",GridZoneDes[(80 + (short)lat) / 8 + 3]);

   /* Determine the number of the MGRS grid zone designator */
   if((CM == 9.0) && (MGRS2[0] == 'V'))
        strcat(MGRS1, "32");
   else sprintf(MGRS1,"%2d",(((180 + (short)CM) / 6) + 1));

   /* Determine the first letter of the 100,000 m area identifier */
   switch(((short)CM + 180) % 18)
   {
    case 3:  sprintf(MGRS3,"%c",GridZoneDes[(short)(Easting / 1e5)]);          break;
    case 9:  sprintf(MGRS3,"%c",GridZoneDes[(short)(Easting / 1e5)+8]);        break;
    case 15: sprintf(MGRS3,"%c",GridZoneDes[(short)(Easting / 1e5)+16]);     break;
    default:  break;
   } /* end switch */

   /* Determine the second letter of the 100,000 m area identifier */
   if((((short)CM + 180) % 12) == 9) toffset = toffset + 5;

   sprintf(MGRS4,"%c",GridZoneDes[(((short)(Northing / 1e5) + toffset) % 20) + 1]);
   tN= Northing - ((short)(Northing / 1e5) * 1e5);
   if(Northing < 0) tN = 1e5 + tN;

   sprintf(NStr,"%5d",(int)tN);
   sprintf(EStr,"%5d",(int)(Easting - ((short)(Easting / 1e5) * 1e5)));

   /* Form the MGRS string */
   strcat(MGRS,MGRS1);  strcat(MGRS,MGRS2);  strcat(MGRS,MGRS3);
   strcat(MGRS,MGRS4);   strcat(MGRS,EStr);    strcat(MGRS,NStr);

   for(j = 0; j <= 14; j++)
       if(MGRS[j] == ' ') MGRS[j] = '0';

} /* end UTM2MGRS */
```

Figure 7.9 Function UTM2MGRS

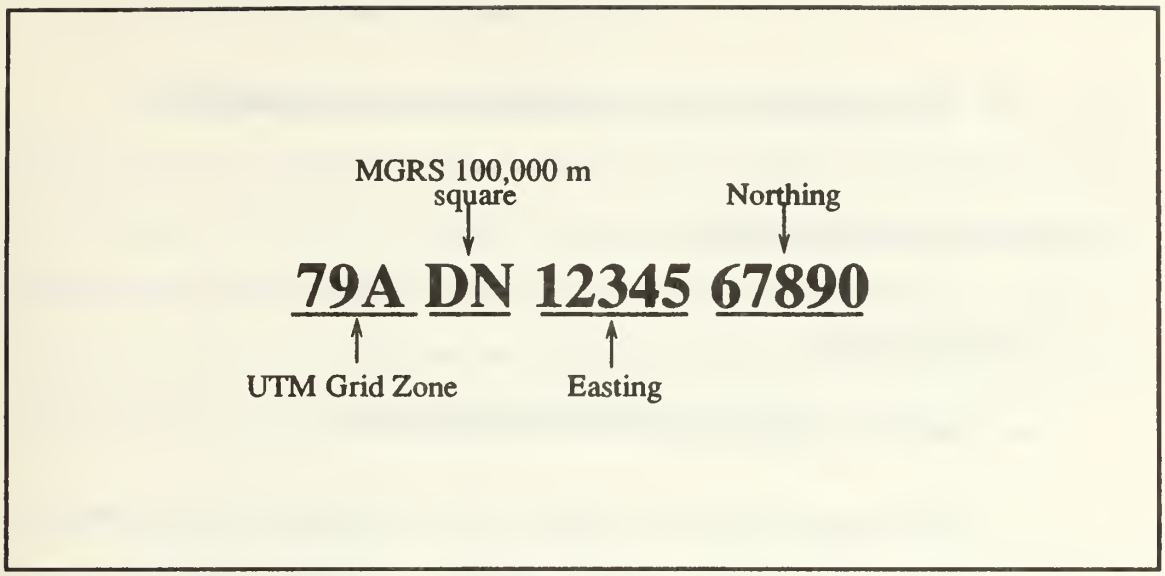


Figure 7.10

C. USE OF MGRS IN FOST

As explained previously, MGRS is the standard means of location in the Field Artillery. Because of this, FOST uses MGRS in almost every area of execution. The use of actual MGRS grids in FOST is one of the primary sources of realism for our training system.

FOST computes the terrain data base MGRS location as soon as the user selects the file. This grid is the base location from which the program calculates all other locations. Using this base grid, the program labels all the map grid lines with either the grid number or 100,000 meter square identifier. FOST computes MGRS grids for all targets and OPs. All FO mission data inputs and calculations are based on the MGRS. MGRS is an integral part of FOST and a critical element to its realistic nature.

VIII. PROGRAM AND PERFORMANCE ASSESSMENT

A. PROGRAM ASSESSMENT

1. System Strengths

a. Real-Time, 3D Training With Moving Simulation

Our prototype trainer demonstrates some of the many ways to produce innovative training systems using today's technology. Computer generated three dimensional graphics provide a realistic training environment that the Field Artillery is not currently exploiting. FOs and their targets become both mobile and potentially interactive. The training ground depicts the future battleground. Computer programs like FOST put the trainer in control.

b. User Friendly

A key concern for any highly technical system is that of user friendliness. FOST's program setup is completely menu driven and self explanatory. There are only two devices a soldier needs to handle: the dials control his vehicle and a mouse makes DMD selections and enables his binoculars. Noncommissioned Officers (NCO) can easily train these skills, which, with today's growing rate of computer literacy, are already familiar to many of our soldiers.

c. Virtually Unlimited Database Selection

With FOST's ability to access DMA data files, soldiers can train on terrain for almost any contingency plan. Trainers have available to them data files that cover virtually the entire globe. The trainers can easily access these data files from either local or remote databases that they can access through networks like the Defense Data Network. Soldiers need not become tired and overfamiliar with the "same old scene"; they can train on the ground on which they will fight.

d. Flexible Training

NCOs can tailor training sessions to individual or even group abilities and adjust the system as the needs of the unit grow or change. With FOST's small space requirements and relatively inexpensive resources, it is readily available to battalion level units. This advantage reduces the need to compete with other units for time with the simulator, eliminating the administrative burden of scheduling and coordinating with outside organizations. Because of the greater availability, NCOs could conduct informal training with soldiers needing more attention. Soldiers could also train by themselves without their supervisor, during their own time. FOST provides an adaptable environment, ready to meet the Army's training needs.

2. Available and Inexpensive Resources

Workstations like the IRIS provide simple, easy to use programming shells that facilitate in-house work. We developed FOST from MPS in less than nine months. Assets are available at NPS, other graduate institutions and at government facilities to continue development on FOST, or similar systems, at minimal cost. Our

prototype illustrates a way we can cut costs in software development by taking advantage of personnel we send to advanced schooling.

B. PERFORMANCE ASSESSMENT

Because FOST is a successor to MPS, we use tests similar to the MPS evaluation scheme as the basis for our performance assessment [Ref 1]. However, FOST is much more graphically and computationally demanding than MPS, so the test results do not exactly correspond. FOST has all the original complexity of MPS, plus enhancements like: the terrain mesh drawing, MGRS conversion and all of the Field Artillery specific additions (e.g. DMD, adjustments, rounds and craters).

We conducted four tests to determine the performance characteristics of FOST (see Figure 8.1). Since, in most cases, the FO's area of responsibility is large, we chose to display attenuated terrain. The zoom angle corresponds to the FO's use of binoculars. The 15 degree angle is the FO's viewing angle when the binoculars are in use and the 55 degree angle is the FO's unaided viewing angle. For the nine vehicle scenario, the FO occupies a jeep and the other eight vehicles are tanks.

C. PERFORMANCE CONCLUSION

FOST runs about two times slower than MPS [Ref. 1]. This decrease in speed from MPS is primarily due to the DMD operation. During data entry FOST switches operation from the 3D display window to the DMD keyboard window, momentarily freezing the 3D action. While this does not seriously degrade system realism, it is by no means optimal performance. Operating FOST on a multi-processor machine would

allow seperate processors to simultaneously control both windows, allowing greater performance.

DISPLAYING ATTENUATED TERRAIN

PLATFORM	ZOOM ANGLE	POLYGONS PER FRAME	FRAMES PER SECOND
TWO VEHICLES	55	960	4.0
TWO VEHICLES	15	668	4.5
NINE VEHICLES	55	1030	3.0
NINE VEHICLES	15	814	3.5

Figure 8.1 FOST Performance Data

IX. SYSTEM LIMITATIONS

Our prototype has some limitations that further development must address. The majority of these limitations are due to our desire to produce a basic trainer within the time constraints of our graduate program. We discuss these limitations and how they detract from the system's training realism.

A. DMD LIMITATIONS

1. Message Formats

Because we have not implemented all the menus, missions and functions included in a real DMD, the DMD is not yet fully operational . Because we directed our main effort towards a functional prototype, we limited the DMD menus to the minimum necessary to conduct a basic grid mission with subsequent adjustments. An actual DMD has approximately 20 different top level message formats ranging from various mission types, (grid, polar, illumination, etc.) to free text. The time investment necessary to implement all these formats would have been greater than the training return at this point in system development. It is important to recognize, however, that many formats are currently not available for training.

2. Shell Fuse Combinations

An actual DMD offers all current artillery shell fuse combinations. FOST offers four shell/fuse combinations: High Explosive/Point Detonating (HE/PD), High Explosive/Variable Time (HE/VT), High Explosive/Mechanical Time (HE/TI) and Im-

proved Conventional Munitions (ICM). We chose these combinations, because in our experience they represent the most typical munitions forward observers request in their calls for fire. Additionally, they are inexpensive to simulate both computationally and graphically. By limiting our program to only these shell/fuse combinations, we reduce our prototype's potential range of training.

3. Mouse vs. Actual DMD Keyboard Input

Because a DMD's keyboard is in alphabetical order rather than a standard QWERTY keyboard order, we chose to implement on-screen input. We recognize that this form of implementation is a double edged sword. With on-screen input, the key locations and names are identical to the actual equipment. The user becomes accustomed to searching the simulated DMD keyboard in the same place as the actual equipment. However, mouse input adds a training artificiality and with each artificiality, training realism is degraded.

B. REALISM LIMITATIONS

1. Terminal Effects

a. Effects On Targets

In order to determine the appropriate number of rounds to shoot at a particular target, Field Artillery fire direction computers use munitions effects tables. Given the type and size of a target, along with the desired percentage of destruction, these tables yield the required number of rounds to produce the effect. The necessary number of rounds increase as the size of the target and desired percentage of destruction increase, or as the target vulnerability decreases. Forward observers are re-

sponsible for accurately identifying target type and size so that the fire direction computer can determine the required number of rounds.

FOST's terminal effects do not take into account data from munitions effects tables. Since our main thrust is to provide the FO visual feedback on the accuracy of his call for fire, we implemented a simple effects scheme. The Army Readiness Training Evaluation Program (ARTEP) 6-400 standard gives effects on target if a round lands within a fifty meter radius [Ref 16]. We use this effects standard, but we destroy our target, as opposed to the ARTEP standard of neutralization, in order to give the FO positive feedback on the accuracy of his mission. This simple scheme slightly degrades the program's training benefits by providing inaccurate effects on the target.

b. Lack of Sound Effects and Smoke Residue

A FO in combat has a statistically short life expectancy; because of the lethality of the weapons he directs the enemy wants to quickly eliminate him. The FO attempts to remain behind cover and concealment as much as possible when adjusting a mission. He often relies on the sound of the projectile exploding to know when to look up. He then makes his adjustment to the target using the lingering smoke, rather than the flash, of the explosion.

FOST lacks the realism of sound and smoke residue. The Iris 4D/70GT we developed FOST on does not have the hardware necessary to generate sound effects. We opted to leave out the smoke effect due to the time limitations of our graduate program. The user must rely on using the burst or the crater as his adjusting point if the round is off target.

c. Smoke And Illumination Projectiles

FOST lacks the capability to train FOs to call for and adjust either obscuration (smoke) or illumination missions. Our primary rational for not including these missions in the prototype is again to avoid degradation of system performance. The Iris 4D/70GT has the hardware to support the simulation of smoke and illumination rounds, however the price of this simulation is a reduction of program speed. We believe the cost of adding realistic transparent smoke or moving local light sources, at this time, is far greater than the training benefits gained.

2. 3D Vehicle Icons Are Of Limited Complexity

While FOST includes the basic array of military vehicles, they are simplistic both in terms of drawing and function. We draw the vehicles generically without regard to nationality or political affiliation, (e.g. Warsaw Pact vs NATO). This lack of differentiation extends to vehicle color; the vehicles are all olive drab without any distinguishing camouflage pattern. The result is, the only way for an FO to distinguish an enemy vehicle from a friendly is whether it is moving towards him or along with him.

Both friendly and enemy vehicles are passive, functioning only as observation posts or targets. Unless the vehicle is the driven vehicle for that workstation, FOST lacks the ability to maneuver these vehicles. Realistically, these vehicles not only maneuver, but engage each other with direct fire weapons. FOST does not provide this capability in its current configuration. We chose to concentrate on the primary utility of the vehicles as targets.

3. At Any Given Time The Current Program Limits Operation To One 10km Area

One of the greatest limitations of FOST's current configuration is the inability to use the entire terrain database, at one time, when in the 3D phase. Although a user can effectively cover an entire 120km x 120km database by selecting different 10km areas, this is not the best method for a training system. Ideally the user should have access to the entire database, regardless of which 10km area he selects as his initial area of operations. Future development must make correction of this limitation a top priority.

4. Absence Of Man-made Or Natural Terrain Features

FOST displays pure terrain. Because it uses DTED data to create the terrain, FOST does not show most of the natural terrain features that exist on the area the data covers. Features such as forests do not appear and rivers and lakes will only appear if they happen to be at Sea Level. FOST also does not show any of the man-made features such as buildings, roads, or railroad tracks. Thus FOST currently gives the FO familiarity with only the topographic elements of the area not the cultural features.

5. Loss Of Some Depth Effect On Flat Terrain

In order to achieve a more realistic three dimensional view of the terrain, we use the mesh drawing technique. This technique smooths out the differences in elevation between data points and when combined with a coloring scheme based on elevation, the terrain appears natural and rolling. A slight problem occurs when there is an expanse of flat terrain, i.e. terrain whose elevation difference does not cause a change

in the coloring scheme. The problem is that FOST looses some of its depth effect on these flat areas when using the mesh technique instead of a checkerboarding technique (see Figures 6.4 and 6.5). We choose to keep the mesh technique in spite of this problem because it is far more realistic than the checker boarding technique.

X. POTENTIAL FOR SYSTEM GROWTH

A. ADDITION OF DIGITAL TERRAIN FEATURES ANALYSIS DATA (DFAD)

The Defense Mapping Agency uses two types of data files to create maps. One type is DTED files, which are the same files that we use in FOST to display elevations. DMA uses the DTED files to provide map elevations. The other type is DFAD; this file provides the cultural information such as roads, buildings, rivers and forests. By combining DTED with DFAD files DMA is able to produce maps which accurately reflect the face of the Earth.

As we note in Chapter IX, one of FOST's limitations is the absence of any cultural information. Although the procedures for drawing 3D buildings, forests and other features are available, it would be extremely difficult for an individual to attempt to place these features prior to each program run. DFAD files provide a way to overcome this limitation accurately.

Modifying FOST to accept and display information from DFAD files provides realistic and easy to update cultural data. In that way, what a user sees on his military map is what FOST displays in 3D. This is a great and much needed improvement to the prototype training system.

The DFAD file structure is very similar to the DTED, however the data is very different. Cultural features have code numbers which identify them by type, along with information pertaining to size in three dimensions. DMA collects DFAD data in

two levels: Level 1 is a generalized description and portrayal of the data, suitable for large scale maps (over 1:200,000); Level 2 is a more detailed portrayal of the data, suitable for small scale maps (1:50,000). For this enhancement we recommend Level 2 data as it contains the detail which is appropriate for Small Unit Training systems like FOST.

We must make it clear that this is by no means a trivial addition to the code. To make this improvement work, an individual must create a new routine to read in the DFAD data following the DTED data. FOST currently uses a single data structure for its terrain drawing routine. This improvement requires integration of the DFAD and DTED data into a single data structure.

It is important for the DFAD file to follow the DTED file, because DFAD does not adjust linear lengths for changes in elevation.* Integration of the files requires a routine to adjust the linear lengths of cultural features due to the elevation changes. Anyone attempting to add DFAD must make every effort to streamline the coding and drawing routines as the cultural data will greatly increase the total polygon count. The total polygon count affects program performance.

This improvement alone constitutes an enormous amount of work. By successfully accomplishing the 3D display of DFAD and DTED data, the utility of all the 3D simulators under research at NPS would be greatly enhanced.

---

*Without adjusting for changes in elevation, problems could occur in 3D drawing. For example, DFAD describes a road between two data points 100 meters apart. If there is no elevation change between the data points then we can draw the road 100m long. If however there is a difference in the elevations the road is longer than 100m.

B. ARTIFICIAL INTELLIGENCE INTEGRATION

1. Expert System Based Tutor

The use and development of expert systems is growing rapidly in the military and FOST is another application that can take advantage of this technology. An expert system tutor using artificial intelligence techniques can integrate directly into the FOST program. This system can provide not only after action feedback to the FO, but advise him on proper conduct of fire. Trainers could then concentrate their attention on weaker soldiers who need personal attention, while their more advanced FOs learn with the tutor.

While the Iris workstations at NPS have Franz Lisp on line, it lacks an easily accessible inference engine mechanism for an expert system. We believe that either Prolog or a networking scheme with a Symbolics workstation would be better suited for coding the expert system.

Prolog and C code easily integrate into one system; most currently available Prolog compilers and interpreters are in the C programming language. Prolog is a powerful reasoning language, suited to forward chaining and database manipulations which teaching systems generally require [Refs. 17, 18]. There are also many prototype expert systems available in Prolog which could serve as models for FOST's expert system. While it is possible to network from a workstation running Prolog, the best option, for efficiency, is the addition of a Prolog compiler on the Iris.

The Symbolics workstations have the KEE expert system development tool available. This tool develops expert systems using a hybrid dialect of Common Lisp which includes object oriented features. KEE is very powerful and is suited for devel-

opment of complex expert systems such as tutors. This option also requires the implementor to develop a networking scheme between the Iris and the Symbolics workstations.

We believe that an expert system based tutor will greatly improve FOST's utility as a military training system. This area will easily provide a student with a challenging thesis topic that has real world utility.

2. Smart And Aggressive Targets

"Smart targets" are another feature that is possible through software programs. Other graduate students at NPS developed an M1 tank simulator using the same basic code and hardware as FOST. Integration of FOST with this system would provide operator controlled targets that not only move, but also shoot back.

An alternative approach to this feature is to use artificial intelligence techniques of path planning and collision avoidance. "Smart" enemy vehicles could autonomously attack and maneuver on your position. Tying all of this together results in a combined arms training system: armor and mechanized infantry firing and maneuvering, artillery providing fire support.

Artificial intelligence languages are specially suited for this purpose. We believe that implementing this feature using either Prolog or Lisp on the Symbolics workstation will be most effective.

C. EXERCISE EVALUATOR FOR FIELD ARTILLERY MUNITIONS EFFECTS

An area of potential growth that should spark a lot of interest is not directly related to training forward observers, but concerns fire support assessment. We briefly mentioned how the US Army Test and Experimentation Command (USATEC) at Fort Ord is using FOST's predecessor to depict and record exercise vehicle maneuvers. By expanding FOST to include firing unit positions, munitions effects tables and other gunnery related parameters, USATEC could integrate FOST into their current system. Evaluation of indirect fire effects would become more accurate. Evaluators could observe effects that FOST realistically simulates on-screen and assess accurate casualties and vehicle losses.

Follow-on work to the original Moving Platform Simulator set up a system to display vehicles in either real time or from a database. The Army could use this work as the model for a Field Artillery exercise evaluator. There is a real world need for an accurate evaluation mechanism of this type. We believe an effort in this area will pay great dividends in the training arena.

D. GRAPHICAL ENHANCEMENTS

1. Vehicle Nationality and Pattern Painting

The total complexity of the generated image is the only limit to graphical enhancements of the current system. In adding new or more complex objects to the 3D display we must keep in mind the trade off between realism and speed. The program must run at an acceptable level, in terms of frames per second, to be useful as a training system.

An individual desiring to add more realism to FOST with little, if any, effect on the performance should redesign the vehicles to accurately reflect both NATO and Warsaw Pact equipment. The drawing algorithms are simple. With some graph paper and attention to detail about the surface normals, we can easily display the vehicles of the various nations.

Another improvement to FOST's realism is in the area of vehicle coloring. NATO and Warsaw Pact nations paint their combat vehicles with a camouflage pattern to increase survivability by decreasing the chance of detection. FOST's vehicles lack this pattern painting and therefore are less realistic.

2. Smoke And Illumination Rounds

The absence of the obscuration (smoke) and illumination missions is a distraction from FOST's otherwise considerable utility as a training system. The simulation of these missions primarily involves work with lighting, shading, and transparency models. By taking advantage of the Iris 4D/70GT's lighting functions, a student can easily add these features at a cost in performance. The challenge becomes to add these missions with a minimum cost to the system in terms of frames per second.

The drawing and lighting functions necessary for one to add these features are similar to the functions currently in FOST. Because the facilities, both hardware and software are available in the Iris 4D/70GT we consider these additions secondary in our work.

3. Projection For Group Training

Enrichments are not limited to just additional software. Supplemental hardware is available to provide facilities for simultaneous team training beyond the net-

working capabilities present in FOST. Commercially available high resolution video projection systems allow FOST to provide a group training facility similar to the TS-FO, but with all the advantages of three dimensional computer simulation.

E. DIRECT INTERFACE TO ACTUAL EQUIPMENT

The best way for a FO to become proficient on the equipment we expect him to use in combat is for him to use it often in training. FOST currently fails to allow the user to work on the Digital Message Device, substituting an on screen DMD. A fix of this deficiency is the creation of an interface which allows fire mission input from a DMD to the program.

The DMD is a electronic device which transmits digital message packets via a radio or telephone line to a fire direction computer. Since the DMD operates in a digital format, the program's translation of the package structure is relatively straightforward. The DMD requires a hardware interface to link it to the Iris 4D/70GT. The program then operates with out the on-screen DMD, but with a real DMD. The program receives DMD message packets in a fashion similar to packets other workstations send in networking mode.

This interface holds great potential for further improvement of FOST. We believe that this is worthy of further research. Adding the DMD involves not only format translation but also networking and hardware interfacing.

LIST OF REFERENCES

1. Fichten, Mark A. and Jennings, David H., *Meaningful Real-Time Graphics Workstation Performance Measurements*, M. S. Thesis, Naval Postgraduate School, Monterey, California, December 1988.
2. Naval Postgraduate School, Technical Report NPS52-89-004, *Meaningful Real-Time Graphics Workstation Performance Measurements,* Fichten, Mark A., Jennings, David H., and Zyda, Michael J., November 1988.
3. Smith, Douglas B., and Streyle, Dale G., *An Inexpensive Real-Time Interactive Three-Dimensional Flight Simulation System*, M.S. Thesis, Naval Postgraduate School, Monterey, California, June 1987.
4. Oliver, Michael R., and Stahl, David J., *Interactive, Networked, Moving Platform Simulators*, M.S. Thesis, Naval Postgraduate School, Monterey, California, December 1987.
5. Silicon Graphics Inc., *IRIS User's Guide, MEX Window Manager*, Mountain View, California, 1987.
6. Silicon Graphics Inc., *4Sight User's Guide, Volume 1*, Mountain View, California, 1988.
7. Silicon Graphics Inc., *IRIS User's Guide, GT Graphics Library User's Guide*, Mountain View, California, 1987.
8. Winn, Michael and Strong, Randolph, *Moving Platform Simulator II: A Networked Real-time Simulator With Intervisbility Displays*, Master's Thesis, Naval Postgraduate School, Monterey, California, June 1989.
9. Department of the Army, Field Manual Number 6-30, *Observed Fire Procedures,* Government Printing Office, Washington, DC, 17 June 1985.
10. Department of the Army, Field Manual 21-26, *Map Reading and Land Navigation*, pp 4-13 thru 4-16, Government Printing Office, Washington, DC, 1987.
11. Thompson, M. M., *Maps For America*, pp 240-244, Government Printing Office, Washington, DC, 1979.
12. Richardus, Peter and Adler, Ron K., *Map Projections*, pp. 137-143, American Elsevier, 1972.
13. Ressler, Gene and Loomer, Scott A., *Geographic to Geodetic Coordinate Conversions*, Research Computer Program, Department of Geography and Computer Science, United States Military Academy, West Point, New York.

14. Department of the Army, Technical Manual 5-237, *Surveying Computer's Manual*, pp 312-317, Government Printing Office, Washington, DC, 1957.

15. Department of the Army, Technical Manual 5-241-2, *Universal Transverse Mercator Grid Zone to Zone Transformation Tables*, Government Printing Office, Washington, DC, 1957.

16. Department of the Army, Army Training and Evaluation Program 6-400, *The Field Artillery Battalion*, pg 3-138, Government Printing Office, Washington, DC, 1984.

17. Rolston, David W., *Principles of Artificial Intelligence and Expert Systems Development*, pp. 169-178, McGraw-Hill, Inc., 1988.

18. Weiss, Sholom M. and Kulikowski, Casimir A., *A Practical Guide to Designing Expert Systems*, pp 11-13, Rowman & Allanheld, 1984.

19. Department of the Army, Technical Manual Number 11-7440-281-12&P, *Operator's and Organizational Maintenance Manual Including Repair Parts and Special Tools List, Digital Message Device AN/PSG-2A (NSN 7025-01-094-5473)*, Government Printing Office, Washington, DC, 18 May 1982.

20. Lehmann, Charles H., *Analytic Geometry*, pp. 116-125, John Wiley & Sons, Inc., 1942.

21. Graham, Palmer H., John, F. Wallace and Hollis, Cooly R., *Analytic Geometry*, pp. 143-150, Prentice-Hall, Inc., 1936.

INITIAL DISTRIBUTION LIST

1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, CA 93943-5002	2
3.	Dr. Michael J. Zyda Naval Postgraduate School Code 52, Department of Computer Science Monterey, CA 93943-5100	200
4.	CPT William T. Drummond, Jr. 819 Sunrise Ct. Derby, KS 67037	3
5.	CPT Joseph P. Nizolak, Jr. 29 Roberta Dr. Howell, NJ 07731	3
6.	Director of Combat Developments United States Army Field Artillery Center and School Fort Sill, OK 73503	1
7.	COL Gerald E. Galloway, Jr. Department of Geography and Computer Science United States Military Academy West Point, NY 10996	1
8.	Mr. Mike Tedeschi United States Army Test and Experimentation Command Attention: ATCT-TE-TM Fort Ord, CA 93941-7000	1

9. Commander, TRADOC 1
ATTN: ATRM-K (LTC McGarrahan)
Fort Monroe, VA 23651-5000

10. Mr. George Lukes 1
United States Army Engineer Topographic Laboratories
Fort Belvoir, VA 22060-5546

11. LTC Anthony Anconetani 1
Headquarters, Department of the Army
Artificial Intelligence Center
ATTN: CSDS-AI
Washington, DC 20310-0200

Thesis
D7765
c.1

Drummond
A graphics workstation
Field Artillery Forward
Observer Simulation
Trainer.